Open University of Catalonia

Doctoral Thesis

From Metaheuristics to Learnheuristics: Applications to Logistics, Finance, and Computing

Author:
Laura Calvet Liñán

Thesis committee:
Dr. Ángel A. Juan Pérez
Dr. Carles Serrat i Piè
Dr. David Masip Rodó

A thesis submitted in fulfillment of the requirements for the degree of Doctor of Network and Information Technologies

in the

Internet Computing & Systems Optimization (ICSO)
Internet Interdisciplinary Institute (IN3)

May 3, 2017

OPEN UNIVERSITY OF CATALONIA

Abstract

Internet Interdisciplinary Institute (IN3)

Doctor of Network and Information Technologies

From Metaheuristics to Learnheuristics: Applications to Logistics, Finance, and Computing

by Laura Calvet Liñán

A large number of decision-making processes in strategic sectors such as transport, production and finance involve $\mathcal{NP}$-hard problems. Trends such as globalization make systems larger and more complex. Frequently, these problems are characterized by high levels of uncertainty and dynamism. Metaheuristics have become predominant methods for solving challenging optimization problems in reasonable computing times. However, they frequently assume that the inputs, the objective functions, and the set of optimization constraints are deterministic and known in advance. These constitute strong assumptions that lead to work on oversimplified versions of real-world problems. As a consequence, the solutions obtained may have a poor performance when implemented. Simheuristics integrate simulation into metaheuristics to solve stochastic optimization problems in a natural way. Similarly, learnheuristics combine statistical learning and metaheuristics to tackle optimization problems in dynamic enviroments, where inputs may depend on the structure of the solution.

From a methodological perspective, the main contributions of this thesis are the design of learnheuristics and a classification of works hybridizing statistical / machine learning and metaheuristics. It discusses the potential of learnheuristics in a number of fields and studies two specific cases. The first is a routing problem in which the depots are heterogeneous, in terms of their commercial offer, and customers show different willingness to consume depending on how well the assigned depot fits their preferences. Thus, different customer-depot assignment maps lead to different customer-expenditure levels. Regression models are employed to capture the relationship between each customer's willingness to spend as a function of several variables, including the assigned depot as well as other customer's features (age, gender, etc.). The second case describes a vehicle routing problem where each customer's demand depends on the order in which the customers are visited. Moreover, several applications are presented in transport, production, finance, and computing. In the first field, the smart design of routes, including capacitated depots and vehicles, are addressed analyzing stochastic demands, and sustainability indicators. Moreover, the waste collection problem and a routing problem with a heterogeneous fleet, asymetric costs and site-dependency are studied. In the production arena, the optimization of jobs' sequences under stochasticity, considering multiple production lines and a common deadline, is discussed. Strategies to invest on risky assets are proposed and assessed. Finally, the parameter fine-tuning of metaheuristics and the effect of the number of agents and the computing time on metaheuristics' performance are investigated.

OPEN UNIVERSITY OF CATALONIA

Resum

Internet Interdisciplinary Institute (IN3)

Doctor of Network and Information Technologies

From Metaheuristics to Learnheuristics: Applications to Logistics, Finance, and Computing

by Laura CALVET LIÑÁN

Un gran nombre de processos de presa de decisions en sectors estratègics com el transport, la producció i les finances impliquen problemes $\mathcal{NP}$-difícils. Tendències com la globalització fan que els sistemes siguin cada cop més grans i complexos. Sovint, aquests problemes es caracteritzen per alts nivells d'incertesa i dinamisme. Les metaheurístiques s'han convertit en mètodes molt populars per resoldre problemes d'optimització difícils en temps de càlcul raonables. No obstant això, sovint assumeixen que els inputs, les funcions objectiu, i el conjunt de restriccions d'optimització són deterministes i coneguts. Aquests constitueixen supòsits forts que obliguen a treballar en versions simplistes de problemes del món real. Com a conseqüència, les solucions poden conduir a resultats pobres quan s'apliquen. Les simheurístiques integren la simulació a les metaheurístiques per resoldre problemes d'optimització estocàstica d'una manera natural. Anàlogament, les learnheurístiques combinen l'estadística amb les metaheurístiques per fer front a problemes d'optimització en entorns dinàmics, on els inputs poden dependre de l'estructura de la solució.

Des d'un punt de vista metodològic, les principals contribucions d'aquesta tesi són el disseny de les learnheurístiques i una classificació dels treballs que combinen l'estadística / l'aprenentatge automàtic i les metaheurístiques. La tesis discuteix el potencial de les learnheurístiques en un conjunt de camps i estudia dos casos específics. El primer és un problema d'enrutament en el qual els magatzems són heterogenis, en termes de la seva oferta comercial, i els clients mostren diferents disposicions a consumir en funció de com el magatzem assignat s'ajusti a les seves preferències. Per tant, diferents mapes d'assignació de clients a magatzems condueixen a diferents nivells de despesa. Es fan servir models de regressió per representar la relació entre la disposició de cada client per consumir com una funció de diverses variables, incloent el magatzem assignat, així com característiques d'altres clients (edat, gènere, etc.). El segon cas descriu un problema d'enrutament on la demanda de cada client depèn de l'ordre en què es visiten aquests. D'altra banda, es presenten diverses aplicacions en el transport, la producció, les finances, i la informàtica. En el primer camp, el disseny intel·ligent de rutes, incloent magatzems i vehicles amb capacitat limitada, s'aborda analitzant demandes estocàstiques i indicadors de sostenibilitat. A més a més, el problema de la recol·lecció de residus i un problema d'enrutament amb una flota heterogènia, i costos asimètrics i dependents del lloc s'estudien. En l'àmbit de la producció, es discuteix l'optimització de seqüències de tasques considerant estocasticitat, múltiples línies de producció i una data límit. Es proposen i avaluen estratègies per invertir en actius de risc. Finalment, s'investiguen la selecció de valors dels paràmetres de les metaheurístiques i l'efecte de la quantitat d'agents i del temps de càlcul en el rendiment de les metaheurístiques.

OPEN UNIVERSITY OF CATALONIA

Resumen

Internet Interdisciplinary Institute (IN3)

Doctor of Network and Information Technologies

From Metaheuristics to Learnheuristics: Applications to Logistics, Finance, and Computing

by Laura CALVET LIÑÁN

Un gran número de procesos de toma de decisiones en sectores estratégicos como el transporte, la producción y las finanzas implican problemas $\mathcal{NP}$-difíciles. Tendencias como la globalización hacen que los sistemas sean cada vez más grandes y complejos. Con frecuencia, estos problemas se caracterizan por altos niveles de incertidumbre y dinamismo. Las metaheurísticas se han convertido en métodos muy usados para resolver problemas difíciles de optimización en tiempos de computación razonables. Sin embargo, suelen asumir que los inputs, las funciones objetivo y el conjunto de restricciones de optimización son deterministas y se conocen de antemano. Estas fuertes suposiciones conducen a trabajar en versiones simplificadas de problemas del mundo real. Como consecuencia, las soluciones obtenidas pueden tener un pobre rendimiento cuando se implementan. Las simheurísticas integran simulación en metaheurísticas para resolver problemas de optimización estocástica de una manera natural. De manera similar, las learnheurísticas combinan aprendizaje estadístico y metaheurísticas para abordar problemas de optimización en entornos dinámicos, donde los inputs pueden depender de la estructura de la solución.

Desde un punto de vista metodológico, las principales aportaciones de esta tesis son el diseño de las learnheurísticas y la clasificación de los trabajos que combinan estadística / aprendizaje automático y metaheurísticas. La tesis discute el potencial de las learnheurísticas en una serie de campos y estudia dos casos específicos. El primero es un problema de enrutamiento en el que los almacenes son heterogéneos, en términos de su oferta comercial, y los clientes muestran una disposición diferente de consumir dependiendo de lo bien que el almacén asignado se ajuste a sus preferencias. Por lo tanto, diferentes mapas de asignación de clientes a almacenes conducen a diferentes niveles de consumo. Se utilizan modelos de regresión para representar la relación entre la disposición de cada cliente a gastar en función de varias variables, incluyendo el almacén asignado, así como las características de otros clientes (edad, género, etc.). El segundo caso describe un problema de enrutamiento de vehículos en el que la demanda de cada cliente depende del orden en que se visitan los clientes. Además, se presentan varias aplicaciones en transporte, producción, finanzas e informática. En el primer campo, el diseño inteligente de rutas, incluyendo almacenes y vehículos con capacidad limitada, se aborda analizando demandas estocásticas e indicadores de sostenibilidad. También se estudia el problema de la recolección de residuos y un problema de enrutamiento con una flota heterogénea, y costes asimétricos y en función del sitio. En el ámbito de la producción, se analiza la optimización de secuencias de tareas considerando estocasticidad, múltiples líneas de producción y un plazo común. Se proponen y evalúan estrategias para invertir en activos de riesgo. Finalmente, se investigan el ajuste de parámetros de metaheurísticas y el efecto del número de agentes y el tiempo de computación en el rendimiento de las metaheurísticas.

Acknowledgements

First of all I would like to thank my supervisors Dr. Angel A. Juan and Dr. Carles Serrat for their guidance, support, innovative ideas and timely feedbacks. I owe a great debt of gratitude for their patience and inspiration. I am also grateful to my advisor Dr. David Masip for his supervision, help and motivation. I feel highly privileged to have worked with him.

I would also like to thank my colleagues in the Internet Computing & Systems Optimization (ICSO) research group and collaborators for their support both in research and personal life. My warmest greetings to Enoc Martínez, Jana Doering, Dr. Javier Panadero, Lorena Reyes, Dr. Sara Hatami, Dr. Carlos Quintero, Aljoscha Gruler, Carla Talens, Stephanie M. Alvarez, Dr. Adela Pages and Dr. Jésica de Armas.

I would like to extend my sincere acknowledgment to Dr. Madeleine Lopeman and Dr. Guillermo Franco, and Dr. Renatas Kizys for their hospitality during the research stays in Dublin and Portsmouth, respectively.

I am also grateful to those who offer me their help in the most crucial stage, the admission to the doctoral program. I would like to express my very warm gratitude to Dr. Joan Manuel Marquès for his encouragement, and Dr. Daniel Riera and Dr. Àlex Sànchez for their support.

Likewise, I want to convey my gratitude to the Doctoral School of the Open University of Catalonia (UOC) for all the help and guidance provided. This thesis would have never been possible without the support of the Internet Interdisciplinary Institute (IN3) that funded me.

Finally, I would like to thank my family and friends (present and gone, humans and dogs) for their tireless support and endless love. They have brought me here.

To all of them I dedicate this thesis.

Contents

List of Figures

List of Tables

List of Abbreviations

ABC	**A**rtificial **B**ee **C**olony
ACO	**A**nt **C**olony **O**ptimization
AIS	**A**rtificial **I**mmune **S**ystems
ALNS	**A**daptive **LNS**
ARP	**A**rc **R**outing **P**roblem
ARPO	**A**lgorithm for **R**ich **P**ortfolio **O**ptimization
ASP	**A**lgorithm **S**election **P**roblem
BA	**B**at **A**lgorithm
BDS	**B**est **D**eterministic **S**olution
BKS	**B**est **K**nown **S**olution
BSS	**B**est **S**tochastic **S**olution
CBR	**C**ase-**B**ased **R**easoning
CEF	**C**onstrained **E**fficient **F**rontier
COP	**C**ombinatorial **O**ptimization **P**roblem
COPDI	**COP** with **D**ynamic **I**nputs
CVRP	**C**apacitated **VRP**
CWS	**C**larke and **W**right **S**avings
DE	**D**ifferential **E**volution
DOE	**D**esign **O**f **E**xperiments
DPCS	**D**istributed and **P**arallel **C**omputing **S**ystems
DFSP	**D**istributed **F**lowshop **S**cheduling **P**roblem
DPFSP	**D**istributed **PFSP**
DPFSP-ST	**DPFSP** with **S**tochastic **T**imes
DSS	**D**ecision **S**upport **S**ystem
EA	**E**volutionary **A**lgorithm
EDA	**E**stimation of **D**istribution **A**lgorithm
EITP	**E**nhanced **I**ndex **T**racking **P**roblem
FA	**F**irefly **A**lgorithm
FSP	**F**lowshop **S**cheduling **P**roblem
GA	**G**enetic **A**lgorithm
GP	**G**enetic **P**rogramming
GRASP	**G**reedy **R**andomized **A**daptive **S**earch **P**rocedure
GVRP	**G**reen **VRP**
HBMO	**H**oney **B**ees **M**ating **O**ptimization
HS	**H**armony **S**earch
HSAVRP	**H**eterogeneous **S**ite-dependent **A**symmetric **VRP**
HoSAVRP	**H**omogeneous **S**ite-dependent **A**symmetric **VRP**
HSAVRP-SD	**HSAVRP** with **S**tochastic **D**emands
ILS	**I**terated **L**ocal **S**earch
IG	**I**terated **G**reedy
IPTS	**I**nstance-specific **P**arameter **T**uning **S**trategies
IRP	**I**nventory **R**outing **P**roblem
ITP	**I**ndex **T**racking **P**roblem
IWO	**I**nvasive **W**eed **O**ptimization
LNS	**L**arge **N**eighborhood **S**earch
LR	**L**ogistic **R**egression
LRP	**L**ocation **R**outing **P**roblem
MDVRP	**M**ulti-**D**epot **VRP**
MDVRP-HD	**MDVRP** with **H**eterogeneous **D**epots
MDVRP-SD	**MDVRP** with **S**tochastic **D**emands

MCS	**M**onte **C**arlo **S**imulation
MFN	**M**ulti-layer **F**eedforward **N**etwork
MLR	**M**ultiple **L**inear **R**egression
MOEA	**M**ulti-**O**bjective **EA**
MS	**M**ulti- **S**tart
NN	**N**eural **N**etwork
OR	**O**perations **R**esearch
PCA	**P**rincipal **C**omponents **A**nalysis
PCS	**P**arameter **C**ontrol **S**trategies
PFSP	**P**ermutation **F**lowshop **S**cheduling **P**roblem
PFSP-ST	**PFSP** with **S**tochastic processing **T**imes
PM	**P**opulation-based **M**etaheuristic
POP	**P**ortfolio **O**ptimization **P**roblem
PSO	**P**article **S**warm **O**ptimization
PSP	**P**arameter **S**etting **P**roblem
PTS	**P**arameter **T**uning **S**trategies
PVRP	**P**ollution **VRP**
RAT	**R**adio **A**ccess **T**echnologies
RMP	**R**isk **M**anagement **P**roblem
RVRP	**R**ich **VRP**
SA	**S**imulated **A**nnealing
SAM	**S**uccessive **A**pproximations **M**ethod
SOM	**S**elf-**O**rganizing **M**aps
SCOP	**S**tochastic **COP**
SM	**S**ingle-solution based **M**etaheuristic
SMEs	**S**mall and **M**edium **E**nterprises
SPOP	**S**tochastic **POP**
SR-GCWS-CS	**S**imulation in **R**outing via the **G**eneralized **CWS** heuristic with **C**ache and **S**plitting
SS	**S**catter **S**earch
SVM	**S**upport **V**ector **M**achine
TS	**T**abu **S**earch
TSP	**T**ravelling **S**alesman **P**roblem
UEF	**U**nconstrained **E**fficient **F**rontier
VNS	**V**ariable **N**eigborhood **S**earch
VRP	**V**ehicle **R**outing **P**roblem
VRP-SD	**VRP** with **S**tochastic **D**emands
WCP	**W**aste **C**ollection **P**roblem
WCP-SW	**WCP** with **S**tochastic **W**aste levels

Chapter 1

Introduction

This chapter introduces the thesis. Its sections are: motivation, main goal and original contributions, and dissertation outline.

1.1 Motivation

Metaheuristics constitute a heterogeneous family of algorithms designed to solve a high number of complex combinatorial optimization problems (COPs) without having to deeply adapt them to each problem (Boussaïd et al., 2013). They represent the first recourse for $\mathcal{NP}$-hard problems that are required to be solved in real time. Their quickness and flexibility to address realistic and rich (i.e., complex) problems are two significant advantages in comparison to exact methods. However, they can not guarantee optimal solutions, but tend to provide near-optimal ones. As synthesized in Talbi (2009), "optimization is everywhere; optimization problems are often complex; then metaheuristics are everywhere". Certainly, metaheuristics are highly popular among researchers and companies, and may be found in a large number of fields. For instance, their use is frequent in: routing, scheduling, telecommunications, machine learning, cryptology, etc. The first works on heuristics (more experience-based procedures) were written in the 1940s (in particular, Polya, 1945, is considered to be the first), but metaheuristics started to be widely used in the 1970s and 1980s. Since then, their importance in operation research (OR) has rapidly increased.

Fields such as logistics (including both transport and production logistics), finance and computing are strategic sectors for all developed economies. A number of complex decision-making processes in real-life related applications can be modeled as COPs (Faulin et al., 2012). All theses problems are $\mathcal{NP}$-hard in nature, which leads to the use of metaheuristics. In logistics, road transport is key for the efficient flow of goods in supply chains. The multi-depot vehicle routing problem (MDVRP) represents a non-trivial extension of the classical vehicle routing problem (VRP) combining assignation and routing issues. In the MDVRP with heterogeneous depots (MDVRP-HD), the customers show different willingness to consume depending on how well the assigned depot fits their preferences. The MDVRP with stochastic demands (MDVRP-SD) allows demands to follow probability distributions, either theoretical o empirical ones. Also in the routing arena, the interest of the waste collection problem (WCP) and the WCP with stochastic waste levels (WCP-SW) is growing due to the expansion of cities and the relevance of the negative externalities of this service. In fact, there is a growing concern for environmental and social impacts of routing activities in general, which calls for the design of routes based on sustainability indicators. Regarding production, task scheduling is present in the elaboration of products, the design of timetables, etc. The permutation flowshop scheduling problem (PFSP) is a classical problem, which usually aims to find the permutation of jobs that minimizes the total makespan, considering different machines and related restrictions. For instance, the PFSP with stochastic processing times (PFSP-ST) has been extensively studied during the last decade. Frequently, there is a product composed of several components that need to be independently processed before a deadline, when they have to be assembled and the product delivered. This problem is called distributed permutation flowshop scheduling problem with stochastic times (DPFSP-ST). In finance, investments drive the economic growth and social welfare of countries. Metaheuristics are becoming key methodologies for addressing a wide range of problems in this field. For instance, the portfolio optimization problem (POP) consists in selecting a subset of risky assets from a portfolio and setting the weight of the investment of each asset in order to minimize the portfolio's variance for a given required rate of return. Most works fail to account for stochastic returns and covariances, rendering them unrealistic in the presence of heightened uncertainty in financial markets. On the contrary, the

stochastic POP (SPOP) deal with returns and covariances modelled as random variables. Finally, computing includes a large number of procedures that may be optimized. Some examples are the parameter-fine tuning on metaheuristics, and the analysis of the effect of the number of agents and the maximum computing time on metaheuristics' performance.

Nowadays, there are two important trends in the literature on metaheuristics. The first sustains the original ideas of metaheuristics: the practical usefulness and logic of simple methodologies relying on local searches (see e.g., Gardi et al., 2014). In contrast, the second encompasses hybrid methodologies, which benefit from the advantages of each component. Indeed, simplicity is an important criteria to assess an algorithm. It facilitates the correct implementation of the algorithm by researchers and companies. However, there are many reasons why an hybrid algorithm may be required: *(i)* to obtain better results in terms of objective function values and/or computational times; and *(ii)* to deal with more realistic and richer problems. For instance, most matheuristics (Maniezzo et al., 2009) fall into the first case, solving a subproblem with an exact method. Talbi (2013) presents the combinations of metaheuristics and: *(i)* complementary metaheuristics; *(ii)* exact methods; *(iii)* constraint programming; and *(iv)* machine learning. While the author discusses interesting ideas, the number of works cited is low and the proposed classification is neither based on works nor on applications, but is a very general framework usable for all the combinations mentioned. In the context of hybrid algorithms, there is another powerful type which combines metaheuristics and simulation, so called simheuristics (Juan et al., 2015a). The framework has been developed during the last years and aims to reduce the lack of works addressing stochastic COPs (SCOPs) (Bianchi et al., 2009). Indeed, the literature has studied some of these problems but usually relying on analytical approaches making hard assumptions or complex approaches. It is crucial to address the stochasticity of these problems, since it is present in most real-life applications. For instance, traveling times greatly depends on factors which are difficult to predict (and so their effects) such as the weather, road works, accidents, etc. Similarly, processing times in scheduling can be affected by machine failures, delays in inputs delivers, etc.

This thesis studies and integrates powerful and well-known methodologies such as simulation, metaheuristics and statistical learning. It presents several works aiming to further explore, test and disseminate simheuristics. An original contribution is the development of learnheuristics combining statistical learning and metaheuristics to deal with COP with dynamic inputs (COPDIs), i.e., problems in which the inputs depend on the solution. A number of applications are analyzed, focusing on the fields previously mentioned: routing, production, finance, and computing. A graphical representation of the main methodologies and applications is shown in Figure 1.1.

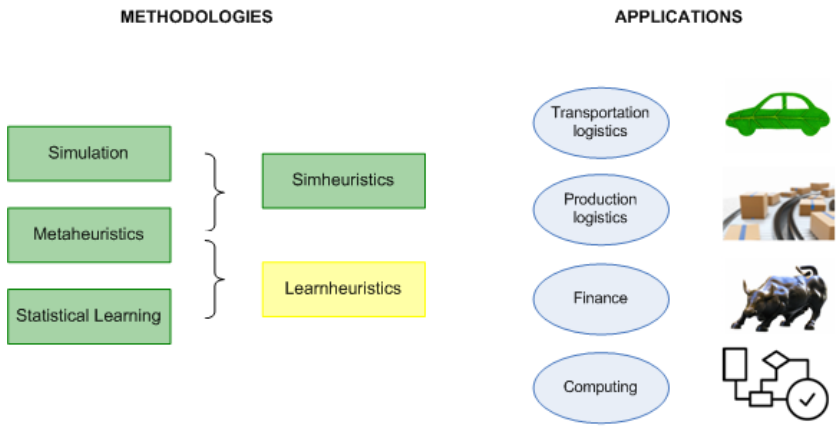


FIGURE 1.1: Scheme of key methodologies and applications of this thesis.

1.2 Main goal and original contributions

The main goal of this research is to explore the advantages of hybrid algorithms combining statistical learning and/or simulation techniques with metaheuristics. It facilitates the design of methodologies able to solve realistic and rich problems, avoiding hard assumptions usually present in the related literature. This general goal has been concreted in the following results and original contributions:

C1. An original approach combining metaheuristics and statistical learning for solving COPDIs, and a general classification and review of works combining statistical learning and metaheuristics.

C2. In the routing arena, efficient metaheuristics for solving the MDVRP-HD, the WCP, powerful simheuristics for addressing the MDVRP-SD, the WCP-SW, and the HSAVRP-SD, and a novel approach considering sustainability indicators.

C3. Regarding production, a simple simheuristic for tackling the DPFSP-ST.

C4. Related to finance, a comprehensive review of works on portfolio optimization and risk management relying on metaheuristics, an efficient metaheuristic and simheuristic for addressing the POP and the SPOP, respectively, and an analysis of the benefits due to diversification of introducing commodity futures in stock portfolios.

C5. Considering computing, a classification and an extensive review of works on the parameter fine-tuning of metaheuristics, a methodology based on clustering techniques and design of experiments (DOE) for the parameter fine-tuning of metaheuristics, and an analysis of the effects of increasing the number of agents and the computing time on the performance of well-known heuristics.

1.3 Dissertation outline

The rest of this thesis is structured in the following three blocks: methodology (chapters 2 to 5), applications (6 to 9), and conclusions, future research, and contributions (10 and 11).

The first block focuses on the existing methodology employed and the pure methodological contributions. In particular, chapter 2 introduces metaheuristics, describing their context, reviewing the main definitions and classifications, and presenting a few popular ones. Chapter 3 is devoted to simheuristics, i.e., the integration of simulation techniques into metaheuristics-based frameworks to deal with SCOPs. Chapter 4 provides a brief definition of statistical learning, highlighting the main branches and methods. Afterwards, chapter 5 puts forward learnheuristics, which combine statistical learning and metaheuristics to address COPDIs.

The block of applications covers problems in transportation, production, finance, and computing. Chapter 6 analyzes five challenging transportation problems: the MDVRP-SD, the MDVRP-HD, the MDVRP with sustainability indicators, the WCP, and the HSAVRP-SD. Different metaheuristics/simheuristics are designed, implemented and validated for them. In the context of production, chapter 7 deals with the DPFSP-ST. Chapter 8 studies optimization problems in finance, presenting a review and focusing on the POP and the SPOP. Next, chapter 9 addresses the parameter fine-tuning of metaheuristics, and discusses issues of parallel computing.

Finally, the last block draws some conclusions, and identifies potential lines of future work in chapter 10, while lists the publications, and presentations in chapter 11. The most relevant contributions may be found at appendices A, B, C and D, gathering journal papers indexed in ISI JCR, journal papers in ISI JCR under review, selected journal papers indexed in Elsevier-Scopus, and selected conference papers, respectively.

Part I

METHODOLOGY

Chapter 2

Metaheuristics optimization

This chapter presents metaheuristics. After introducing them, the main classification criteria are discussed, and biased randomization techniques are presented. Later, the following ones are described: the multi-start, the iterated local search, the simulated annealing and the variable neighborhood search.
It is based on the following journal articles: Calvet et al. (2017), Calvet et al. (submitted[a]), and Calvet et al. (submitted[b]).

2.1 Introduction

OR is a well-established field with a huge and active research community. One of its main goals is to support decision-making processes in complex scenarios, i.e., providing optimal (or near-optimal) solutions to COPs defined by a given objective function and a set of realistic constraints. The number of applications is immense, e.g.: transportation and logistics, finance, production, and telecommunication systems. A noticeable part of the efforts developed by the OR community has focused on developing exact methods to find optimal solutions to a wide range of COPs. When dealing with $\mathcal{NP}$-hard COPs, this usually requires simplifying somewhat the model and/or addressing only small- and medium-sized instances to avoid incurring in prohibitive computing times. Another noticeable part of the efforts has been invested in developing heuristic and metaheuristic approaches that cannot guarantee optimality of the provided solutions but are usually more powerful in terms of the size of the instances they can solve in reasonable computing times (Talbi, 2009). Additionally, these approximated methods are quite flexible, which makes them suitable for tackling more realistic and richer models. While heuristics are simple and fast procedures based on the specific COP being addressed, metaheuristics represent a heterogeneous family of algorithms designed to solve a high number of COPs without having to deeply adapt them to each problem.

Metaheuristics have an enormous number of applications in many fields such as: engineering design, telecommunications, robotics, bioinformatics, system modeling, chemistry, and physics, among many others. A number of them are nature-inspired, include stochastic components, and have several parameters that must be fine-tuned and may interact (Boussaïd et al., 2013). As Feo and Resende (1995) state, the effectiveness of metaheuristics depends upon their ability to adapt to a particular instance problem, avoid entrapment at local optima, and exploit the structure of the problem. The authors also highlight the potential benefit of restart procedures, controlled randomization, efficient data structures, and preprocessing.

2.2 Classification

Many classification criteria have been proposed to differentiate metaheuristics (Talbi, 2009). The most important are highlighted next.

- Memory usage versus memoryless methods.
- Iterative versus greedy. An iterative metaheuristic is built from one (or more) complete solution, which is transformed at each iteration. On the other hand, a greedy algorithm starts from an empty solution and, as the execution proceeds, it is built progressively.

- Deterministic versus stochastic. A deterministic metaheuristic makes deterministic decisions; consequently, using the same initial solution will lead to the same final solution. Whereas with stochastic metaheuristics, one could obtain different solutions.
- Single-solution based search versus population-based search. Single-solution based metaheuristics transform a single solution during their execution. While in population-based metaheuristics, a set of solutions is considered. The first group is exploitation oriented, they intensify the search in local regions. In contrast, population-based metaheuristics are exploration oriented, they allow a better diversification.

Figure 2.1 includes some of the most popular metaheuristics (first works are cited): ant colony optimization (ACO) (Dorigo, 1992), artificial immune systems (AIS) (Farmer et al., 1986), genetic algorithms (GA) (Holland, 1962), greedy randomized adaptive search procedure (GRASP) (Feo and Resende, 1989), iterated local search (ILS) (Martin et al., 1992), particle swarm optimization (PSO) (Kennedy and Eberhart, 1995), scatter search (SS) (Glover, 1977), simulated annealing (SA) (Kirkpatrick, 1984), tabu search (TS) (Glover, 1986), and variable neighborhood search (VNS) (Mladenovic, 1995). They are grouped according to the following criteria: *(i)* single-solution versus population-based metaheuristics (SMs and PMs, respectively); *(ii)* whether they use memory; and *(iii)* whether they are nature-inspired. Circles' size is proportional to the number of Google Scholar indexed articles, from 2006 to 2015, that include the complete name of the specific metaheuristic and "metaheuristics" or "heuristics" in the article (March 15, 2016). The success of the first implementations of metaheuristics aroused the interest of journals in new versions of these methods, which increased the number of authors exploring this topic. Unfortunately, some publications add only marginal contributions to the already existing frameworks (Sörensen, 2015). In this chapter four metaheuristics will be introduced: the multi-start (MS), the ILS, the SA and the VNS. Despite being relatively simple, many state-of-the-art optimization methods are based on them.

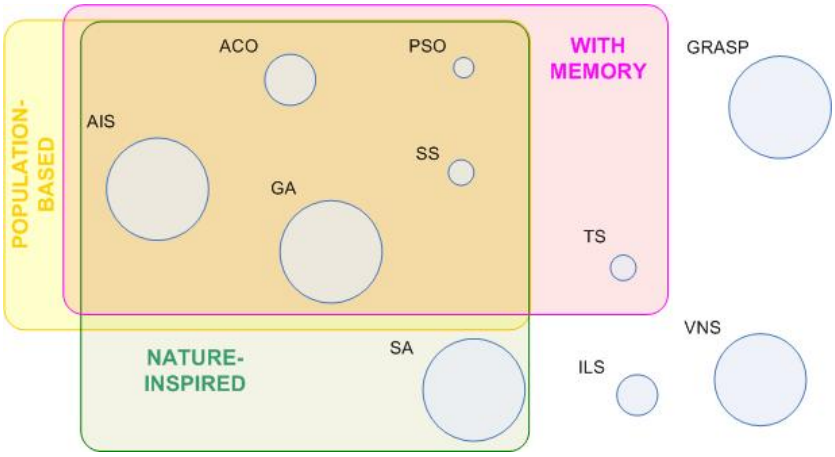


FIGURE 2.1: Main metaheuristics grouped by different criteria.

2.3 Biased randomization

In optimization, a heuristic is defined as a method for building a feasible solution based on an iterative process. At each iteration, the next movement is chosen from a list of potential candidates that has been previously sorted according to a problem-specific criterion. Pure greedy heuristics select the best next element on the short run, aiming to get a good solution at the end. This simple procedure has two drawbacks: *(i)* there is no guarantee of finding an optimal solution; and *(ii)* it is deterministic, so it always returns the same solution. Some pure greedy heuristics are the classical Clarke and Wright savings (CWS) heuristic (Clarke and Wright, 1964) in routing, and the NEH heuristic (Nawaz et al., 1983) in scheduling.

Biased randomization (Grasas et al., submitted) refers to the introduction of randomization in the construction phase and/or the neighborhood search of optimization algorithms. On the one

hand, randomization allows increasing the search area covered by selecting a candidate other than the best next option. On the other hand, biased means that the procedure is not totally random in the sense that not all the elements have the same probability of being selected. In particular, each element is assigned a given probability depending on the criterion of the procedure, the higher the element is in the list, the higher the probability.

Thus, using biased randomization leads to potentially different outputs each time the procedure is executed. Since running a simple heuristic may take a few seconds or less, implementing biased randomization may improve the solution found taking small amounts of time. The assignation of probabilities can be done by using empirical or theoretical probability distributions. There are analytical expressions that allow the quickly generation of random observations from most theoretical distributions. For this reason, these distributions constitute an efficient option. Examples of distributions widely used are the geometric, the triangular, and the log-normal. Algorithm 1 describes the steps required to implement biased randomization.

Algorithm 1 Biased randomization techniques

1: **procedure** BIASED RANDOMIZATION
2: $\mu \leftarrow$ get random number uniformly distributed in $[0, 1)$ given a specific seed
3: $\rho \leftarrow$ get random number from a distribution $PD(parameters, \mu)$
4: $l \leftarrow$ get the ρ element of the sorted list
5: **return** l
6: **end procedure**

Some recent works applying biased randomization are introduced here. The reader interested in a comprehensive review is refered to Grasas et al. (submitted). In the context of smart cities, Mazza et al. (2016) study the use of computation offloading for delegating computing-intensive tasks of smart mobile devices to the cloud. The authors develop a biased-randomized algorithm for solving this assignation problem. Related to real-life transportation activities, Dominguez et al. (2016a) focus on the two-dimensional loading VRP with clustered backhauls, where both delivery and pickup demands are composed of non-stackable items. This work presents a hybrid algorithm integrating biased-randomised versions of vehicle routing and packing heuristics within a large neighbourhood search (LNS) metaheuristic framework. Dominguez et al. (2016b) discuss the two-dimensional loading capacitated VRP with heterogeneous fleet. A MS algorithm based on biased randomization of routing and packing heuristics is proposed. Quintero-Araujo et al. (2016) solve the location routing problem (LRP), which deals with the simultaneous decisions of: *(i)* locating facilities; *(ii)* assigning customers to facilities; and *(iii)* defining routes of vehicles departing from and finishing at each facility to serve the associated customers' demands. A biased-randomized metaheuristic relying on classical heuristics is proposed.

2.4 Multi-start

The MS is a simple metaheuristic consisting of two steps that are alternated for a certain number of global iterations (Algorithm 2). They are: *(i)* generating a solution; and *(ii)* applying a local search (i.e., a procedure to move from one solution to a better one by applying local changes). Each iteration produces a solution, usually a local optimum, and the best one is returned. Muth and Thompson (1963) and Crowston et al. (1963), both focused on scheduling, are considered the first works proposing a MS framework. However, Glover (1977) is the one introducing a local search to improve starting solutions. This author compared procedures for generating starting values for variables and for generating values perturbed from other starting points (known as restarts), and adressed controlled randomization, learning strategies, induced decomposition, and adaptive memory processes (Glover, 1986; Glover, 1989; Glover, 2000).

A comprehensive review on this metaheuristic is provided by Martí et al. (2013). This work describes the origins of the methodology, includes a classification of versions in terms of their use of memory, and introduces adaptive memory programming and the GRASP metaheuristic.

Algorithm 2 Multi-start structure

```
1: procedure MS METAHEURISTIC
2:     repeat
3:         Generate a solution s
4:         s ← LocalSearch(s)
5:     until stopping criterion is met
6: end procedure
```

2.5 Iterated local search

The ILS metaheuristic is a flexible metaheuristic that became very popular at the beginning of this century with the publication of Lourenço et al. (2010), an article that describes and analyzes its framework, reviews the literature, explains the importance of each of the elements involved and the interactions between them, and discusses its relationship with other metaheuristics. Burke et al. (2010) show that the ILS obtains the best average performance among a set of selected metaheuristic approaches in three classical COPs: bin packing, PFSP, and personnel scheduling. The authors also emphasize two main factors for its success: *(i)* an excellent balance between exploration and exploitation by "systematically combining a perturbation followed by local search"; and *(ii)* its simplicity and the reduced number of parameters required, factors that facilitate its quick implementation in practical applications.

The high level architecture of the ILS is shown in Algorithm 3. First, an initial solution is generated, usually employing a random solution or the return of a fast heuristic. Afterwards, a local search is applied to the initial solution. It starts then an iterative process that stops when a termination condition is met; this condition can be based on time, number of iterations or solution converge, among others. Initially, the current solution is perturbed; this process may have memory, i.e., depend on the previous walk (history). It is recommended to implement a random move in a neighborhood of higher order than the one used by the local search algorithm. The following step consists in applying a local search to the perturbed solution. This solution will become the next element of the walk if it passes an acceptance test. Otherwise, one returns to the previous accepted solution. The criteria designed can be adaptive.

An important advantage of this metaheuristic is its modularity, which enables its development without problem-dependent knowledge. However, usually the most knowledge about the problem one introduces, the best performance it gets. The metaheuristic relies on the assumption that local minima are distributed in clusters.

Algorithm 3 Iterated local search structure

```
1: procedure ILS METAHEURISTIC
2:     Generate an initial solution s_0
3:     s* ← LocalSearch(s_0)
4:     repeat
5:         s' ← Perturbation(s*, history)
6:         s* ← AcceptanceCriterion(s*, s', history)
7:     until stopping criterion is met
8: end procedure
```

2.6 Simulated annealing

The SA metaheuristic is a well-established metaheuristic used in both discrete and continuous optimization. It is inspired by the process of physical annealing with solids in which a crystalline solid is heated, and then allowed to cool slowly until it achieves its most regular possible crystal lattice configuration, without crystal defects (Nikolaev and Jacobson, 2010). Similarly, the metaheuristic searches a global solution following this thermodynamic behavior. Algorithm 4 shows the steps in detail. First, a temperature change counter k is initialized to 0. Additionally, a starting temperature t_0 is set, a temperature cooling schedule is designed, and a repetition schedule M_k is stablished, which represents the number of iterations executed at each temperature t_k. An initial

solution s is created, and an outer loop is started. In this loop, a repetition counter m is set to 0. Afterwards, an inner loop starts which repeats the following steps M_k times: *(1)* a new solution s' in the neighborhood of s, $N(s)$, is built; *(2)* the difference between the objective function value of s' and s, $\Delta_{s,s'}$, is calculated; *(3)* if $\Delta_{s,s'}$ is negative (assuming a minimization problem), then s' replaces s, otherwise this replacement is performed with a probability of $exp(-\Delta_{s,s'}/t_k)$; and *(4)* m is incremented by one. After the inner loop stops, k is increased by one, which represents a decrease in the temperature. The criterion applied to decide whether s must be replaced by s' is called Metropolis acceptance criterion (Metropolis et al., 1953). It allows the algorithm to escape from local optima. An interesting overview of this metaheuristic can be found in Suman and Kumar (2006).

Algorithm 4 Simulated annealing structure

```
1:  procedure SA METAHEURISTIC
2:      Build a solution s
3:      Set temperature change counter k = 0
4:      Design a temperature cooling schedule t_k
5:      Design a repetition schedule M_k
6:      repeat
7:          Set repetition counter m = 0
8:          repeat
9:              Build a solution s' ∈ N(s)
10:             Calculate Δ_{s,s'} = f(s') − f(s)
11:             if Δ_{s,s'} ≤ 0 then s ← s'
12:             elses ← s' with probability exp(−Δ_{s,s'}/t_k)
13:             end if
14:             m ← m + 1
15:         until m = M_k
16:         k ← k + 1
17:     until stopping criterion is met
18: end procedure
```

2.7 Variable neighborhood search

The VNS was first proposed by Mladenović and Hansen (1997). Besides being a popular metaheuristic in combinatorial as well as global optimization, it has been used in a wide range of research fields such as scheduling, routing, telecommunications, biology, and artificial intelligence. For extensive reviews on applications the reader is referred to Moreno-Vega and Melián (2008) and Hansen et al. (2010a). In essence, the VNS proposes systematic changes of neighborhood to find a local minimum by intensifying the search, and to escape from the associated valley by diversifying. It relies on three facts: *(i)* a local minimum with respect to one neighborhood structure is not necessarily so for another; *(ii)* a global minimum is a local minimum with respect to all possible neighborhood structures; and *(iii)* for many problems, local minima with respect to one or several neighborhoods are relatively close to each other.

Algorithm 5 shows a simple version of the VNS. Its inputs are the problem instance to solve, the number of neighborhoods considered (K), and the maximum computational time (T). Frequently, K is set to two or three, and the neighborhoods are nested. First, the variable t for measuring the time is initialized at zero. Afterwards, an initial solution is obtained and stored in *currentSol*. An outer loop sets the current neighborhood to the first one, and controls that the time-based constraint is satisfied. Inside, another loop builds and tests new solutions. Within this loop, the current solution is initially shaken (or perturbed), generating a solution from the k-th neighborhood of *currentSol*. The resulting solution is stored in *newSol*, which is then improved by means of a local search. If there is an improvement (i.e., *newSol* is preferred over *currentSol*), *newSol* is copied into *currentSol*, and the current neighborhood is set to the first. This constitutes a descendent phase aimed to find a local minimum. Otherwise, the next neighborhood is analyzed (i.e., k is set to $k + 1$). The inner loop is executed until the last neighborhood is explored (i.e., $k = K$). Finally, *currentSol* is returned.

Algorithm 5 Variable neighborhood search structure

```
1:  procedure VNS METAHEURISTIC
2:      Set K
3:      t ← 0
4:      Generate an initial solution s_0
5:      s* ← s_0
6:      repeat
7:          k ← 1
8:          while k ≤ K do
9:              s' ← Shake(s*, k)
10:             s' ← LocalSearch(s')
11:             if s' ≻ s* then
12:                 s* ← s'
13:                 k ← 1
14:             else
15:                 k ← k + 1
16:             end if
17:         end while
18:     until stopping criterion is met
19: end procedure
```

Chapter 3

Simulation and simheuristics

This chapter presents simheuristics, including a literature review and a discussion of benefits and limitations.
It is based on the following journal articles: Calvet et al. (submitted[a]), Calvet et al. (submitted[b]), and Gruler et al. (2016).

3.1 Introduction

Metaheuristics constitute a powerful approach to tackle COPs, they are indeed highly popular in many research fields. However, these methodologies have been developed considering deterministic problems (i.e., ignoring stochasticity) when, in fact, real-life is plenty of uncertainty. For example, in garbage collection or stocking of vending machines, the demand is not revealed until the place is reached. Other situations in which there is unknown information are flight scheduling and capital management. Unfortunately, the oversimplification of scenarios, i.e., assuming no uncertainty, can lead to poor-quality solutions. This is the reason why there is an increasing interest in considering randomness in COPs (Bianchi et al., 2009).

Simheuristics (Juan et al., 2015a) is an approach combining metaheuristics (in a general sense, i.e., including heuristics, metaheuristics, and exact methods, among others) and simulation (Nance and Sargent, 2002; Borshchev and Filippov, 2004; Gass and Assad, 2005), specially designed to tackle COPs containing stochastic components. These components can be modeled as random variables following either theoretical or empirical probability distributions, and can be located in the objective function (for instance, random processing times) or in the set of constraints (e.g., deadlines that must be met with a given probability).

3.2 Simheuristics

A simheuristic algorithm is a particular simulation–optimization approach oriented to efficiently tackle a COP instance that typically contains stochastic components. These components can either be located in the objective function (e.g., random customers' demands) or in the set of constraints (e.g., deadlines that must be met with a given probability). In particular, the simheuristic approach is aimed at solving COPs of the form:

$$\begin{array}{ll} \text{Min} & f(s) = E[C(s)] \quad \text{or, alternatively,} \\ \text{Max} & f(s) = E[B(s)] \end{array} \tag{3.1}$$

$$\text{subject to:} \quad P(q_i(s) \geq l_i) \geq k_i \quad \forall i \in \{1, 2, \ldots, n\} \tag{3.2}$$

$$h_j(s) \leq r_j \quad \forall j \in \{1, 2, \ldots, m\} \tag{3.3}$$

$$s \in S \tag{3.4}$$

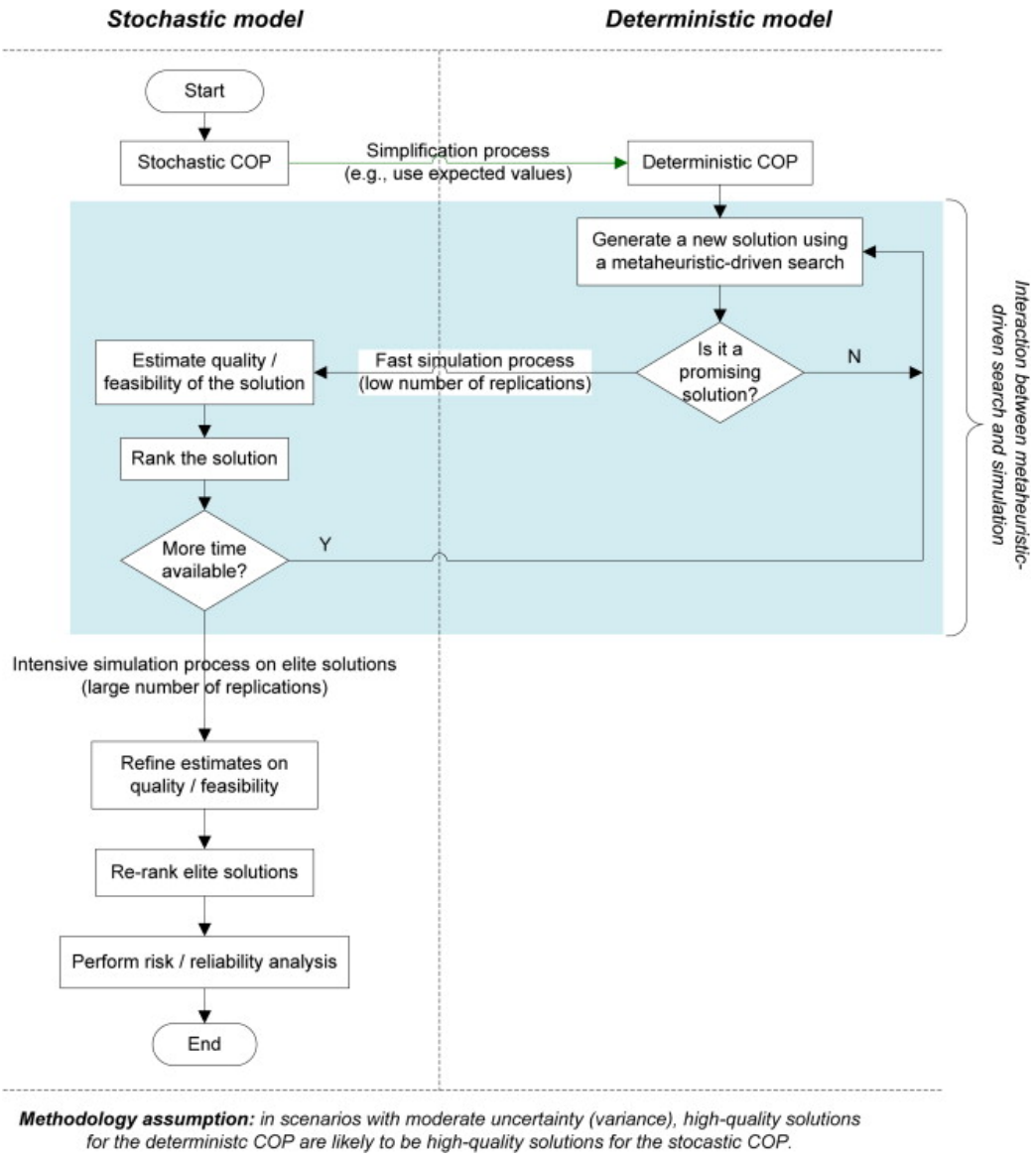
where: (*i*) S represents a discrete space of possible solutions s to the optimization problem; (*ii*) $C(s)$ represents a stochastic cost function (alternatively, $B(s)$ represents a stochastic profit or income function); (*iii*) $E[C(s)]$ represents a probabilistic measure of interest associated with the cost function (e.g., the expected value of $C(s)$); (*iv*) Equations 3.2 represent probabilistic constraints related to the problem (e.g., the probability that the service quality $q(s)$ reaches a given threshold

l is above a user-defined value k); and (v) Equations 3.3 represent typical deterministic constraints in COPs.

The simheuristic approach relies on the assumption that high-quality solutions for the deterministic version of a COP are also likely to be high-quality solutions for its corresponding stochastic version, specially in scenarios with a low or moderate uncertainty (variance).

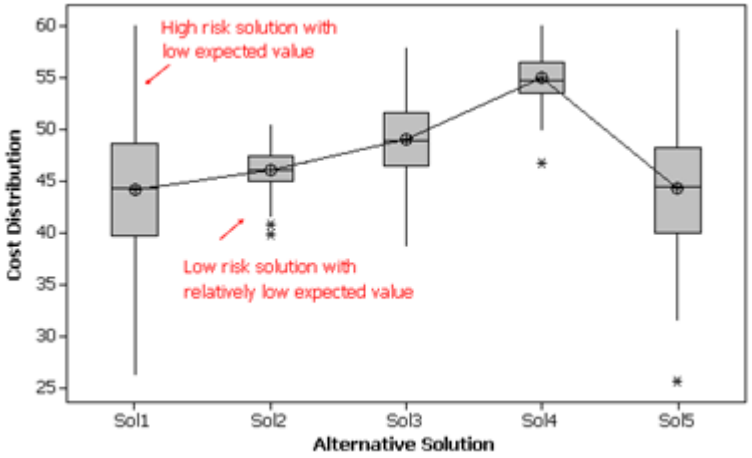
The steps proposed to solve a SCOP instance are described next (see also Figure 3.1). First, a deterministic counterpart of the instance is obtained, for example, by replacing random variables by their expected values. Afterwards, an iterative process is started. It consists in running a metaheuristic-driven algorithm to perform an efficient search inside the solution space associated with the deterministic COP first, and then estimating the quality or feasibility of each of the promising solutions when being considered as solutions of the SCOP instance. These estimations are computed using simulation techniques. The estimated values can be employed to keep a ranked list of the best solutions for the SCOP instance. Once a stopping criteria is met, more accurate estimates are obtained for the best solutions with an intensive simulation process. The advantages of this approach are numerous: it benefits from the extensive literature research related to solve deterministic COPs, and is simple, easy-to-understand and to-implement, efficient, and capable of solving realistic problems.

FIGURE 3.1: Scheme of the simheuristic approach. Source: Juan et al. (2015a)



After identifying a set of high-quality solutions (those that provide the highest average performance), the risk aversion of the decision-maker can be taken into account by performing a risk analysis. It is done by studying the empirical probability distribution functions of the quality and feasibility measures computed during the intensive simulation process. An easy and fast procedure can be to provide a multiple box-plot including the solutions returned by the algorithm or only a subset of them, the non-dominated. It provides information (quartiles and outliers) about the functions. Figure 3.2 shows an example where there is a trade-off between solution risk and expected value.

FIGURE 3.2: Risk analysis of alternative solutions. Source: Juan et al. (2015a).



The described procedure is proposed when simulation is to be used as an evaluation function technique. Nevertheless, simheuristics can also be applied in the context of analytical model enhancement methods. The first case is the most developed so far. An example of the second can be found in Figueira et al. (2013).

Despite the fact that this approach is relatively new, there are many relevant lines of investigation being explored. In Juan et al. (2011b), the authors address the VRP with stochastic demands (VRP-SD) employing safety-stocks to reduce the route failure-risk. The single-period stochastic inventory routing problem (stochastic IRP) with stock-outs is tackled in Juan et al. (2014b). Another routing problem, the arc routing problem (ARP) with stochastic demands is studied by Gonzalez et al. (2016). Juan et al. (2014a) present a methodology to solve the PFSP. In Cabrera et al. (2014), the authors address the SCOP of determining a minimum-cost configuration of non-dedicated resources able to support a specific service while maintaining its availability over a user-defined threshold. Focusing on the home service industry, Fikar et al. (2016) propose a flexible discrete-event driven metaheuristic to deal with dynamic routing and scheduling scenarios using combined trip sharing and walking. It facilitates real-world operations, enabling rescheduling and rerouting.

3.2.1 Benefits

According to Chica et al. (submitted), the most relevant benefits of simheuristics are:

- Embracing reality by a validated simheuristic
  They allow the construction and study of valid complex system models. Indeed, new simulation paradigms can better represent the complexity of reality, and there are computational resources to run demanding simulations for addressing models that are too complicated for analytical models.

- Risk assessment of alternative solutions and sensitivity analysis
  The outputs of the simulations can be employed to generate information about the probability distribution of the quality of each solution. These outputs can also be used to perform a sensitivity analysis, which identifies the parameters having a higher effect on the model. These analyses aim to gain insights into existing or prospective systems, which could lead to better decisions and, as a consequence, to better managerial outcomes.

- System understanding and output analysis
  An innovization process (Deb et al., 2014) consists in analyzing a set of trade-off optimal or near-optimal solutions to decipher useful relationships among problem entities. Visualization methods promote design innovations. An analysis of the input/output variables space of a model may strengthen trust in the solving approach. All these analyses provide a better understanding of the behavior of the optimization and simulation models.

3.2.2 Limitations

The most important limitations are:

- Results are not expected to be optimal
  Metaheuristics do not guarantee the optimality of the solution provided. Additionally, simulation in simheuristics represents a nonlinear complex system which cannot be analytically treated. Thus, simheuristics should be used when simple and flexible methods are needed to address complex problems.

- Additional stakeholders effort is demanded to define the system
  Simheuristics require additional effort when defining the simulation system and analyzing the results.

- More computational resources are required with respect to traditional methods
  Running a simheuristic algorithm requires a high computational effort, which depends on the selected type of simulation paradigm.

Chapter 4

Statistical learning

This chapter introduces statistical learning. It summarizes the basic learning approaches describing the most popular methods and highlighting the main applications.
It is based on the following journal articles: Calvet et al. (2017), Calvet et al. (submitted[c]), and Calvet and Juan (2015).
This work has been presented at the following seminar: De Armas et al. (2016a).

4.1 Introduction

The term "statistics" was originally created in the 18th century to denote the systematic collection of demographic and economic data of a state. Since then, its meaning has been increasingly broadened. Some more updated informal definitions proposed in Hahn and Doganaksoy (2012) are: *(i)* the science of learning from data; *(ii)* the theory and methods of extracting information from observational data for solving real-world problems; and *(iii)* the science of uncertainty. Statistics plays an important role in numerous economics sectors. The world of statistics[1] remarks the most visible: business and industry, health and medicine, learning, research, social sciences and natural resources.

The following subsections present the most basic learning methods classified into supervised and unsupervised learning (Hastie et al., 2009).

4.2 Supervised learning

Supervised learning encompasses a set of procedures for function approximation. Given data pairs $\{x_i, y_i\}\, \forall i = \{1, \ldots, n\}$, in a $(p+1)$-dimensional Euclidean space, there is a function $f(x_i)$ that has a domain equal to the p-dimensional input subspace, and is related to the data via a model such as $y_i = f(x_i) + \varepsilon_i$. The goal of these procedures is to obtain a useful approximation to $f(x_i)$ for all x in some region of $\mathbb{R}^p$.

Functions are estimated to describe a relation between variables, and to predict a response variable based on explanatory variables. Despite the fact that linear models with few explanatory variables are usually robust and powerful, sometimes a more complex model as a neural network can be required. In this case, the user can hardly explain the specific role of each explanatory variable in the model, i.e., the main aim of the model is to make predictions.

It is essential to validate a model before using it. This is done by splitting the dataset into two subsets: a training set, containing the data pairs used to build the model, and the test set, which is used to assess its performance. This split should be random, in order to obtain two representative subsets. Sometimes, it is required to have three subsets: a training, a validation and a test set. The validation test is employed to determine the best model between a set, or to estimate a model-specific parameter like the number of hidden units in a neural network or the parameter that determines the shrinkage penalty in a ridge regression. Often, specially with high-dimensional data ($p >> n$), it is undesirable or unfeasible to perform those splits. Then, a common procedure consists in applying cross-validation, a method for estimating the prediction error of a model based on the following steps: *(1)* generate a given number of disjointed training

[1] The world of statistics is a global network of more than 2.350 organizations worldwide committed to increasing public awareness of the power and impact of statistics, nurturing statistics as a profession, and promoting the development of probability and statistics. Its official web is: www.worldofstatistics.org.

sets from the dataset and define the corresponding test sets (all data pairs except those included in the associated training set); *(2)* for each training set, build a model and compute the prediction error with the corresponding test set; and *(3)* estimate the prediction error of the model as the average. The most popular methods of supervised learning are introduced below.

- Linear regression
  A linear regression model assumes that the relationship between the response variable and the explanatory variables is linear. This relationship is modeled through an error variable ε_i, which is an unobserved random variable. The corresponding model is:

  $$y_i = \beta_0 + \sum_{j=1}^{p} \beta_j x_{ij} + \varepsilon_i \qquad \forall i = \{1, \ldots, n\}$$

- Tree-based methods
  Tree-based methods partition the feature space with splits, and fit a simple model for each subset of data. There are methods both for regression and classification.

- Neural networks
  Neural networks (NNs) extract linear combinations of explanatory variables as derived features, and model a response variable as a non-linear function of these features. Hidden layers are layers between the input and the output layers. Increasing the number of hidden layers and/or hidden neurons adds complexity and improves computational capacity. Having too few hidden neurons, the model might not have enough flexibility to capture the non-linearities in data. NNs tend to have many weights, which might cause problems of overfitting. Weight decay is a method of regularization to prevent it. This method adds a penalty to the error function that shrinks the weights toward zero. A tuning parameter allows weighting the penalty in the error function.

  There are several types of NN, both in supervised and unsupervised learning. They may be employed for prediction, classification, clustering, and ranking. The most popular is the feed-forward NN for prediction in supervised learning.

- Support vector machines
  The basic support vector machine (SVM) algorithm employs a training subset with a binary response variable to build a model capable of assigning new observations into one category. The algorithm constructs a hyperplane so that categories are separated by the widest margin possible. The model may include penalizations in case observations are not separable. This method may perform non-linear classification by employing the kernel trick, and mapping the explanation variables into high-dimensional spaces.

4.3 Unsupervised learning

In unsupervised learning, there is a set of observations $x_i, \forall i = \{1, \ldots, n\}$, of a random p-vector X having joint density $Pr(X)$. The aim is to infer the properties of this probability density. The most popular methods are introduced below.

- Cluster analysis
  Cluster analysis consists in grouping a collection of observations into subsets or clusters, such that those within each cluster are more closely related to one another than those assigned to different clusters. The grouping is based on the definition of similarity / dissimilarity between two observations. The dissimilarity between two clusters is defined by the linkage. The most used types are: complete, single, average, centroid, and medoid. The result of the clustering highly depends on the linkage selected.

  Hierarchical clustering is an approach that aims to build a hierarchy of clusters. The related strategies fall into two types: agglomerative and divisive. The first group starts with each observation being considered a cluster, and pairs of clusters are merged as one moves up the hierarchy. On the other hand, in the divisive approach all observations start in a cluster, and splits are done recursively as one moves down the hierarchy.

- Self-organizing maps
  A self-organizing map (SOM) is a type of NN. It produces a low-dimensional and discretized representation of the input space of the dataset, which is called map. An important characteristic of this NN is that preserves the topological properties of the input space by employing a neighbourhood function. A SOM represents a mapping from the input space to the map space with a lower dimension.

- Principal components analysis
  Principal components analysis (PCA) consists in transforming a dataset to a new coordinate system by applying orthogonal linear transformation in such a way that the greatest variance by some projection of the data comes to lie on the first coordinate or first principal component, the second greatest variance on the second coordinate, and so on. The number of principal components is less or equal to the number of original variables. PCA is usually performed by eigenvalue decomposition of the covariance or correlation matrix of the original data.

There is a third method so-called semi-supervised learning (Chapelle et al., 2006), which employs both labeled and unlabeled observations (i.e., not all have associated an output value). It is extremely powerful for problems in which large amounts of unlabelled observations are available, and only a few of them can be manually labelled. Typical examples are visual object recognition, where milions of untagged images are publicly available, or natural language processing.

Chapter 5

Learnheuristics: statistical learning and metaheuristics

This chapter reviews works combining statistical learning and metaheuristics, and proposes a hybrid approach for tackling optimization problems with dynamic inputs.
It is based on the following journal articles: Calvet et al. (2017).
This work has been presented at the following seminar: **Calvet16n***; Calvet (2015).*

5.1 Introduction

The OR community shows a growing interest in coping with increasingly challenging COPs, such as SCOPs and dynamic COPs (in which some of the problem inputs evolve over time). This might be due to several factors, including: *(i)* the rich characteristics of real-life problems frequently faced by modern companies in sectors such as logistics and transportation (Caceres et al., 2014); *(ii)* the technological development; *(iii)* the availability of vast amounts of Internet-based data; and *(iv)* a shift to a more data-driven culture. During the last years, hybrid approaches have been extensively employed due to their success when dealing with realistic problems, among others: those combining different metaheuristics (Talbi, 2013), matheuristics (i.e., metaheuristics combined with mathematical programming) (Maniezzo et al., 2009), and simheuristics.

The hybridization of metaheuristics with statistical learning is an emerging research field. In this context, the main contributions of this chapter are: *(i)* providing a classification on works combining metaheuristics with statistical learning; and *(ii)* proposing a novel 'learnheuristic' framework, combining a heuristic-based constructive procedure with statistical learning, to deal with COPDIs. In these problems, the inputs are deterministic (i.e., non-stochastic) but, instead of being fixed in advance, they vary according to the structure of the solution (i.e., they change as the solution is being constructed following a heuristic-based iterative process). In this sense, these COPDIs represent an extension of the classical deterministic COPs in which all inputs are given in advance and are immutable. An example of such a COPDI is given next for illustrative purposes. Suppose there is a set of heterogeneous radio access technologies (RATs) that provide pay-per-use services to a group of users. Each user has to be assigned to just one RAT, and each RAT can serve only a limited number of users. The goal is to maximize the total benefit, which depends on the customers' demands. Several scenarios may be described based on the nature of the demands (Figure 5.1): *(i)* they are deterministic and static; *(ii)* they contain some degree of uncertainty but can be modeled as random variables or using fuzzy techniques; and *(iii)* they are dynamic in the sense that they depend on the solution characteristics (e.g., the number of users connected to the same RAT, which has an effect on the service quality and, therefore, on the customers' demands). While the first case corresponds to a classical deterministic COP, the second case introduces a level of uncertainty that usually requires the use of stochastic programming, simulation-optimization, or fuzzy methods. Focusing on the third case, the learnheuristic algorithms have a learning mechanism that updates the input values as the solution is iteratively constructed using the heuristic logic (Calvet et al., 2016d).

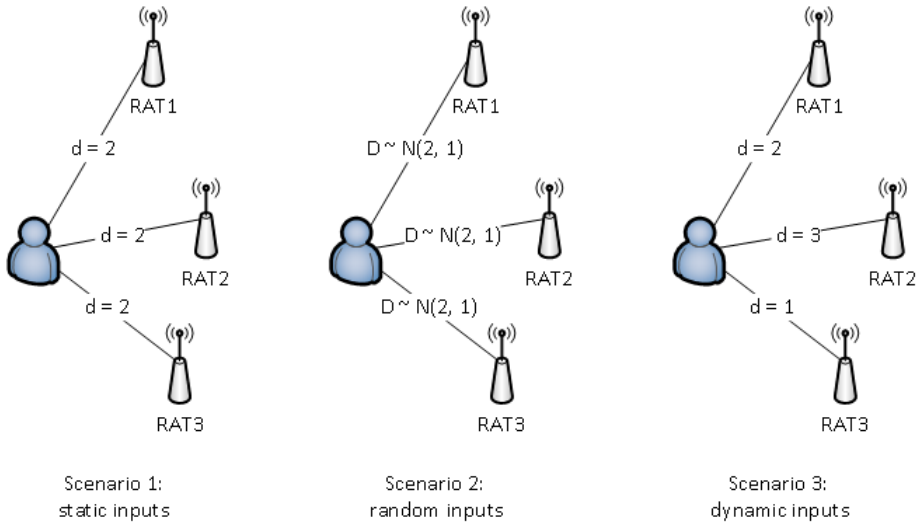


FIGURE 5.1: Type of problem according to the nature of the inputs.

5.2 Related reviews

The existing literature analyzing the hybridization of metaheuristics and statistical learning may be mainly divided into two groups: works were statistical learning is employed to enhance metaheuristics, and those in which metaheuristics are used to improve the performance of statistical learning techniques.

Regarding the first group, there are several works providing overviews. For instance, the emergence of hybrid metaheuristics is studied in Talbi (2013), which includes the combination of metaheuristics and: *(i)* complementary metaheuristics; *(ii)* exact methods; *(iii)* constraint programming; or *(iv)* statistical learning. The author distinguishes between low-level hybridizations, in which a given internal function of a metaheuristic is replaced by another optimization method, and high-level hybridizations, where the different optimization methods are self-contained. In a second phase, these algorithms can be further classified into relay (where techniques are applied one after another) or teamwork hybridization. Jourdan et al. (2006) describe applications of data mining techniques to help metaheuristics. A survey on the integration of statistical learning in evolutionary computation can be found in Zhang et al. (2011). The work presented in Corne et al. (2012) gathers the synergies between OR and data mining, highlighting three benefits of employing data mining in OR: *(i)* increasing the quality of the results; *(ii)* speeding up algorithms; and *(iii)* selecting an algorithm based on instance properties. This chapter builds on the classification in Jourdan et al. (2006) and extends it by proposing more categories and analyzing a higher number of works. Works are classified into specifically-located hybridizations (where statistical learning is applied in a specific procedure) and global hybridizations (in which statistical learning has a higher effect on the metaheuristic design).

Similarly, there are a few reviews on metaheuristics used to improve the performance of statistical learning techniques. For instance, Freitas (2008) focuses on two evolutionary algorithms (EAs), namely GAs and genetic programming (GP), and discusses their application to discovery of classification rules, clustering, attribute selection and attribute construction. Corne et al. (2012) analyze the role of OR in data mining discussing the relevance of exact methods, heuristics and metaheuristics in supervised classification, unsupervised classification, rule mining and feature selection. More recently, Dhaenens and Jourdan (2016) provides an overview of the use of optimization in Big Data focusing on metaheuristics. The book introduces the role of metaheuristics in clustering, association rules, classification, and feature selection in classification. Building on these reviews, here the literature works are arranged into the following categories: classification, regression, clustering, and rule mining.

5.3 Statistical learning for enhancing metaheuristics

This section describes works employing statistical learning for enhancing metaheuristics. They are first grouped into specifically-located hybridizations and global hybridizations.

5.3.1 Specifically-located hybridizations

The ***fine-tuning of metaheuristic parameters*** is known to have a significant effect on the algorithm performance. However, this issue is not always properly addressed and many researchers still continue selecting parameter values by performing exhaustive testing or copying values recommended for similar instances or problems. Basically, there are three approaches:

1. *Parameter control strategies* (De Jong, 2007) apply a dynamic fine-tuning of the parameters by controlling and adapting the parameter values during the solving of an instance. The main types of control are: *(i)* deterministic, which modifies the parameter values by some deterministic rule; and *(ii)* adaptive, which employs feedback from the search. For instance, there are works relying on fuzzy logic (Jeong et al., 2009), SVMs (Zennaki and Ech-Cherif, 2010), and linear and SVM regression (Lessmann et al., 2011).

2. *Parameter tuning strategies* assume that the algorithms are robust enough to provide good results for a set of instances of the same problem with a fixed set of parameter values. Frequently, researchers focus on a subset of the instances and analyze their fitness landscapes. Popular techniques are: response surface (Gunawan et al., 2013), logistic regression (Ramos et al., 2005), and tree-based regression (Bartz-Beielstein et al., 2004).

3. *Instance-specific parameter tuning strategies* present characteristics from the previous approaches. While the parameter values are constant as in the second approach, they are specific for each instance as in the first one. These strategies employ a learning mechanism able to return recommended sets of parameter values given a number of instance features. Techniques employed are: Bayesian networks (Pavón et al., 2009), case-based reasoning (CBR) (Pereira et al., 2013), fuzzy logic (Ries et al., 2012), linear regression (Caserta and Rico, 2009), and NNs (Dobslaw, 2010).

Typically, metaheuristics generate their ***initial solutions*** randomly, using design of experiments (Leung and Wang, 2001), or via a fast heuristic. There are also works employing statistical learning techniques. For instance, some of them apply CBR to initialize GAs (Ramsey and Grefenstette, 1993; Louis and McDonnell, 2004; Li et al., 2011c), while others explore the use of Hopfield NNs (Yalcinoz and Altun, 2001). In De Lima et al. (2008) the authors suggest using the Q-learning algorithm in the constructive phase of a GRASP and a reactive GRASP metaheuristics. In this line, the hybridization of data mining and the GRASP metaheuristic is discussed in Santos et al. (2008).

In real-life applications it is common to find objective functions and constraints that are computationally expensive to ***evaluate*** (Lim et al., 2010; Tenne and Goh, 2010). In these cases, it is required to build an approximation model to assess solutions employing polynomial regression (Zhou et al., 2005), NNs (Adra et al., 2005; Pathak et al., 2008), SVMs (Yang et al., 2009), Markov fitness models (Brownlee et al., 2010), kriging (Díaz-Manríquez et al., 2011) or radial basis functions (Regis, 2014), for example. Some authors combine their use with that of real objective functions (Rasheed and Hirsh, 2000; Zhou and Zhang, 2010). A survey on model approximation in evolutionary computation may be found in Jin (2005). Another option to reduce evaluation costs is to evaluate only representative solutions. Following this idea, Yoo and Cho (2004) apply fuzzy clustering, while Jin and Sendhoff (2004) use clustering techniques and NNs ensembles.

Regarding ***population management***, many authors attempt to extract information from solutions already visited and employ it to build new ones, aiming to explore more promising search spaces. A number of works rely on the Apriori algorithm (to identify interesting subsolutions) (Dalboni et al., 2003; Santos et al., 2005; Ribeiro et al., 2006; Santos et al., 2006) or on CBR (Louis, 2003). Another important issue in PMs is the population diversity, since maintaining it may lead to better performances. The most common technique for promoting diversity is clustering analysis. In Streichert et al. (2003), for instance, individuals in a GA are separated in different sub-populations based on their features and only those in the same cluster compete for survival. The selection operator is applied independently to each cluster.

The search of a metaheuristic may be improved by introducing knowledge in ***operators*** such as mutation or crossover operators in PMs. For example, Michalski (2000) design a class of evolutionary computation processes called learnable evolution model (LEM), which uses symbolic learning methods to create rules that explain why certain individuals are superior to others. These rules are then employed to create new populations by avoiding past failures, using recommendations or generating variants. In Jourdan et al. (2005), this class is extended to address multi-objective problems.

Some statistical learning techniques have been used as ***local searches***. For instance, Gaspar-Cunha and Vieira (2004) employ a multi-objective EA (MOEA) combined with an inverse NN. The authors test their approach on a set of benchmark bi-objective functions. A similar approach is suggested in Adra et al. (2005) to be applied to an aircraft control system design application.

5.3.2 Global hybridizations

A few works have attempted to ***reduce the search space*** in order to make more effective and efficient searches. Statistical learning techniques used are: clustering techniques (Hu and Huang, 2004; Senjyu et al., 2005; Barreto et al., 2007; Adibi and Shahrabi, 2013), NNs (ChangYoon and Way, 2001; Marim et al., 2003) and PCA (Auger and Hansen, 2005).

The ***algorithm selection problem*** (ASP) aims to predict the algorithm from a portfolio that will perform best, employing a given set of instance features. Its framework was proposed by Rice (1976), where it was applied to partial differential equation solvers. More recently, Smith-Miles (2009) presents it in the context of optimization algorithms. Kanda et al. (2011) design an approach to select the best optimization method for solving a given travelling salesman problem (TSP) instance. Initially, 14 TSP properties and the performance values obtained with each metaheuristic analyzed (GRASP, TS, SA and GA) are stored. Then, a rank of metaheuristics is determined by using a multi-layer perceptron network. Several network architectures are assessed. In Smith-Miles et al. (2014), the authors construct a methodology to compare the strengths and weaknesses of a set of optimization algorithms. First, the instance space is generated. This step includes selecting a subset of features providing a good separation of easy and hard instances. Afterwards, classification techniques are used to identify the regions where an algorithm performs well or poorly. The experiment is carried out with 8 algorithms for solving the graph coloring problem.

According to Burke et al. (2010), ***hyperheuristics*** may be described as search methods or learning mechanisms for selecting or generating heuristics to solve computational search problems. Typically, these methods do not aim to obtain better results than problem-specific metaheuristics, but to be able to automate the design of heuristic methods and/or deal with a wide range of problems. The authors propose a classification taking into account the following dimensions: *(i)* the nature of the heuristic search space (either heuristic selection or generation); and *(ii)* the feedback, since hyperheuristics may learn (following online or offline learning strategies) or not. A comprehensive survey on hyper-heuristics may be found in Burke et al. (2013). Reinforcement learning is highly popular in methodologies selecting heuristics employing an online learning strategy (e.g., see Berberoğlu and Uyar, 2010). In Asta and Ozcan (2014) an apprenticeship learning hyperheuristic is proposed for vehicle routing. Taking a state-of-the-art hyperheuristic as an expert, the authors follow a learning approach that yields various classifiers, which capture different actions that the expert performs during the search. While this approach relies on a C4.5 algorithm, in Tyasnurita et al. (2015) it is improved by using a multilayer perceptron.

During the last decades, a new trend in optimization has emerged based on ***cooperative strategies***. It consists in combining several algorithms/agents to produce a hybrid strategy in which they cooperate in parallel or sequentially. Communication among them can be either many-to-many (direct) or memory-based (indirect). Agents may share partial or complete solutions and models, among others. It is broadly accepted that strategies based on agents with unrestricted access to shared information may experiment premature convergence. Commonly, there is an agent that coordinates the search of the others, organizing the communication. For example, Cadenas et al. (2009) develop a centralized hybrid metaheuristic cooperative strategy, where knowledge is incorporated into the coordinator agent through fuzzy rules. These rules have been defined from a knowledge extraction process applied to the results obtained by each metaheuristic. The strategy is tested on the knapsack problem, employing a TS, a SA, and a GA. In Martin et al. (2016), a cooperative strategy relying on different metaheuristic / local search combinations is put forward.

The architecture makes use of two types of agents: the launcher and the metaheuristic agent. Each metaheuristic agent continuously adapts itself according to a cooperation protocol based on reinforcement learning and pattern matching. This proposal is tested on the PFSP and the capacitated VRP (CVRP).

There are several ***new metaheuristics*** based on learning procedures. Most rely on the fact that a set of pseudo-optimal solutions may be considered a sample drawn from an unknown probability distribution. This distribution may be estimated by employing a selected set of promising solutions and used to generate new solutions. A review of these metaheuristics, called estimation of distribution algorithms (EDAs), can be found in Pelikan et al. (2002). These metaheuristics have been employed in a wide range of fields such as routing (Euchi, 2014; Wang et al., 2015), scheduling (Ceberio et al., 2012), and nutrition (Gumustekin et al., 2014).

5.4 Using metaheuristics to improve statistical learning

Metaheuristics have been extensively employed to improve statistical learning tasks. Briefly, some of the most successful approaches are reviewed in the supervised learning topic, both in classification and regression, and in the unsupervised learning topic, including clustering and rule mining.

In ***classification***, metaheuristics have been mainly applied for feature selection, feature extraction and parameter fine-tuning. Escalante et al. (2016) suggest that the bags of visual words algorithm could be improved when non linear combinations of weighted features obtained with GP are considered. The approach is successfully applied to the object recognition field, learning both the weights of each visual word (feature) and the non linear combination of them. Fernández-Caballero et al. (2010) present a multi-classification algorithm relying on multi-layer perceptron NN models. In order to obtain high levels of sensitivity and accuracy (which may be conflicting measures), a Pareto-based multi-objective optimization methodology based on a memetic EA is proposed.

In ***regression***, the use of statistical learning is typically related to the training of complex regression models. Neuroevolution is an emergent field which employs EAs to train NNs. Thus, Yao (1999) provides a literature review on elements evolved: connection weights, architectures, learning rules, and input features. In Stanley and Miikkulainen (2002), the authors develop the neuroevolution of augmenting topologies (NEAT) method, which evolves topologies and weights at the same time. Carvalho et al. (2011) present a methodology to find the best architecture of a NN using metaheuristics. The authors tested the following ones: generalized extremal optimization, VNS, SA, and canonical GA.

Regarding ***clustering***, centroid models are based on an $\mathcal{NP}$-hard optimization problem (thus, only approximated solving methods such as metaheuristics may be employed). For instance, Shelokar et al. (2004) use ACO to cluster objects, obtaining faster results in terms of the number of objective functions evaluations. Gene clustering is performed in Banu and Andrews (2015), where a comparative study is presented based on the following metaheuristics: GA, PSO, cuckoo search and levy flight cuckoo search. More recently, Ferone et al. (2016) present a GRASP metaheuristic for biclustering of gene expression data. The reader can find more details in the applications of metaheuristics to unsupervised learning in these surveys: Hruschka et al. (2009) and Kurada et al. (2013).

Related to ***rule mining***, Freitas (2002) presents data mining tasks and paradigms, and describes the application of GAs and GP for rule discovery, and EAs for generating fuzzy rules. After modeling association rules discovery as an optimization problem, Khabzaoui et al. (2004) explore the use of a GA to obtain associations between genes from DNA microarray data. Noticing that most approaches tend to seek only frequent rules, Khabzaoui et al. (2008) propose a multi-objective approach combining a GA and exact methods to discover interesting rules in large search spaces.

5.5 Learnheuristics

The learnheuristic framework aims at solving COPs in which the model inputs (either located in the objective function or in the set of constraints) are not fixed in advance. Instead, these inputs might vary in a predictable way according to the current status of the partially-built solution at

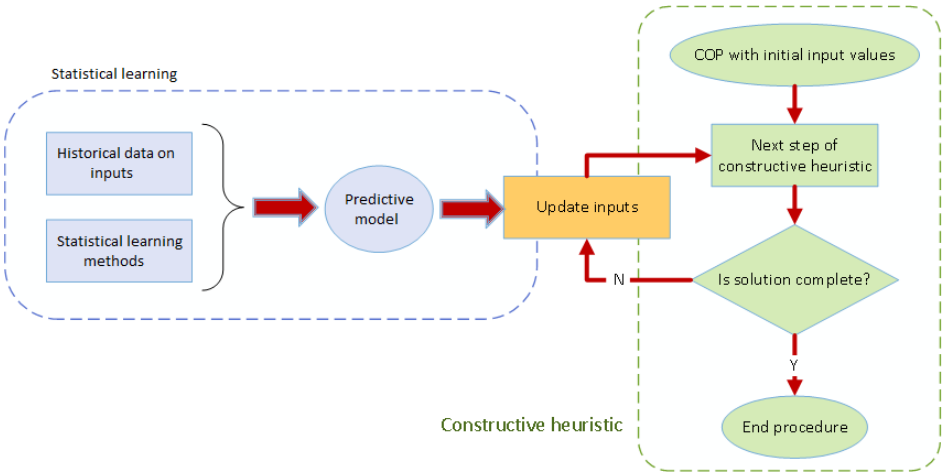


FIGURE 5.2: Basic scheme of a learnheuristic framework.

each iteration of the constructive heuristic. More formally, these problems might be represented as follows:

$$\begin{array}{ll} \text{Min} & C(s, I_{OF}(s)) \quad \text{or, alternatively,} \\ \text{Max} & B(s, I_{OF}(s)) \end{array} \tag{5.1}$$

$$\text{subject to:} \quad Q_j(s, I_C(s)) \leq r_j \quad \forall j \in J \tag{5.2}$$

$$s \in S \tag{5.3}$$

where: *(i)* S refers to a discrete space of possible solutions s; *(ii)* $C(s)$ represents a cost function (alternatively, $B(s)$ represents a benefits function); *(iii)* $I_{OF}(s)$ and $I_C(s)$ refer to inputs in the objective function or the constraints, respectively; and *(iv)* Equations 5.2 represent a set of constraints. Thus, the aim of this type of problems is to minimize a function of costs (or, alternatively, maximize a function of benefits) subject to a number of constraints. The novel characteristic is that inputs in the objective function and/or the constraints may depend on the solution structure, which makes them to be dynamic as the partially-built solution evolves, and not fixed in advance.

In order to deal with these COPDIs, the use of a learnheuristic framework is proposed as explained next (see Figure 5.2). Initially, historical data on different system states (e.g., different assignments of users to RATs) and their associated inputs (e.g., users' demands observed for the corresponding assignments) are employed to generate predictive models. Then, these models are iteratively used during the heuristic-based constructive process in order to obtain updated estimates of the problem inputs (e.g., users' demands) as the structure of the solution (e.g., users-to-RAT assignment map) varies. Eventually, once the construction process is finished, a complete solution is generated. Without the use of the learning mechanism, the heuristic-based construction process will not take into account the variations in the inputs due to changes in the solution structure, which will lead to sub-optimal solutions.

Algorithm 6 contains a more detailed description of the basic learnheuristic framework. Notice that the main loop iterates over a list of elements that are provided by the constructive heuristic (e.g., next user-to-RAT assignment). At each iteration, the algorithm evaluates the current status of the partially-built solution, makes use of the predictive model to update the problem inputs according to this status, and follows the heuristic logic to take another solution-building step based on the new problem inputs.

As any other heuristic procedure, the aforementioned learnheuristic approach can be integrated into a more complex metaheuristic framework. For instance, it can be easily integrated into MS, GRASP, or ILS frameworks. In order to do so, the learnheuristic algorithm may be combined with biased-randomization strategies as the ones proposed in Juan et al. (2011b).

Algorithm 6 Learnheuristic algorithm.

Learnheuristics(*historicalData*, *inputs*)
% *historicalData*: historical data on different system states and their associated inputs
% *inputs*: problem instance
  model ←buildPredictiveModel(*historicalData*)
  sol ← *empty*
  while (*sol* is not completely built) **do** % *iterative learning-heuristic process*
    inputs ←updateInputs(*model*, *inputs*, *sol*)
    sol ← nextHeuristicStep(*inputs*, *sol*)
  end while
  return *sol*

Algorithm 7 Learnheuristic algorithm based on the MS metaheuristic.

Multi-start(*historicalData*, *inputs*, *distribution*, *maxTime*)
% *distribution*: probability distribution and parameters for the biased-randomization process
% *maxTime*: maximum computing time allowed
  initInputs ← *inputs* % *copy of initial inputs*
  elapsedTime ← 0
  initTime ← *currentTime*
  bestSol ← biasedRandLearnheuristic(*historicalData*, *inputs*, *distribution*)
  inputs ← *initInputs* % *reset inputs*
  while (*elapsedTime* ≤ *maxTime*) **do**
    newSol ← biasedRandLearnheuristic(*historicalData*, *inputs*, *distribution*)
    newSol ← localSearch(*newSol*)
    if (cost(*newSol*) ≤ cost(*bestSol*)) **then**
      bestSol ← *newSol*
    end if
    inputs ← *initInputs* % *reset inputs*
    elapsedTime ← *currentTime*−*initTime*
  end while
  return *bestSol*

5.6 Applications

This section provides a series of examples in which the use of learnheuristics might facilitate the solving process of more realistic and rich models.

- ***Transportation:*** In the transportation area and, in particular, in VRPs and ARPs, inputs such as the customers' demands might be dynamic in the sense that they might depend upon the delivery time and whether or not certain time-windows are satisfied. It is a function of the solution structure, e.g., the order in which the customers are visited, the number and type of vehicles employed, etc. Similarly, the traveling times, which affect the distribution cost, might also be dynamic and dependent on the solution structure, specially in large cities where traffic jams occur frequently.
- ***Logistics:*** As discussed in Calvet et al. (2016d), the assignment of customers to certain distribution centers might have a significant effect on the customers' willingness to spend (i.e., on their demands). Therefore, in realistic facility location problems and similar ones, modelers might have to face dynamic inputs influenced by the shape of the solution (i.e., which facilities are open and how customers are assigned to them).
- ***Production:*** In scheduling problems, for instance, processing times of jobs into machines might not be fixed but, instead, they may be a function of the order in which they are processed by the machine (e.g., due to 'fatigue' issues or to breaks). A similar situation can happen in project scheduling, where some working teams might be more efficient than others and assigning them to a given sub-project could cause the delay of others.
- ***Finance:*** In problems such as portfolio optimization, the covariance matrix that measures the risk associated with each pair of assets could also be a function of the current portfolio structure (i.e., which other assets are already included and which percentage of investment has been assigned to each of them). Likewise, the expected return for each asset might depend on the current composition of the portfolio. This dynamic behavior of the inputs can be extended to different risk-management problems which include some sort of portfolio optimization.

5.7 Example

This section describes a simple numerical experiment based on a VRP in which each customer's demand will depend on the order in which the customer is visited. For each customer, its initial demand value is an upper-bound of the real demand. In other words, this value will be valid only if the customer is visited by a vehicle as the first stop in its route. Then, as the position in which the customer is visited increases, the customer's demand will be reduced. Therefore, the use of a constructive heuristic to solve the VRP considering the initial demands as fixed inputs will overestimate the real demands. This, in turn, will lead to higher costs, since the number of routes employed to satisfy the real demands will be higher than necessary. Likewise, vehicles will be carrying more load than strictly required. On the contrary, if the real customers' demands are predicted based on their position inside a route, then each route might be able to cover additional customers and the total distance-based costs will be reduced.

In order to compare both cases, the CWS heuristic have been applied to a random instance belonging to the well known benchmarks for the VRP, particularly to the instance P-n70-k10 (http://neo.lcc.uma.es/vrp/wp-content/data/instances/Augerat/P-VRP.zip). On the one hand, fixed demands are considered, i.e., the original demands are used to obtain the solution through the heuristic in the standard way. On the other hand, a predictive model is created to calculate dynamic demands in order to apply a learnheuristic algorithm. In this case, for illustrative purposes, the following linear regression model has been considered:

$$d = \max\{k_1 \cdot d_0,\ d_0 - k_2 \cdot d_0 \cdot (p-1)\} \tag{5.4}$$

where d is the predicted demand of a given customer, d_0 is the initial demand of the same customer, k_1 and $k_2 \in (0, 1)$, and p is the position order in the route of the aforementioned customer. In particular, k_1 and k_2 are set to 0.20 and 0.05, respectively.

The regression model aims at predicting a customer's demand taking into account the position in which the customer is served in the route, so that the demand decreases as the position increases or until a certain demand lower-bound is reached. Thus, each time the heuristic performs a step, incorporating a new customer in a route or moving a customer from one route to another, the customer's demand is predicted and updated according to its new position in the corresponding route. As mentioned before, the total demand in a route is limited by the capacity of the vehicle. Therefore, this prediction affects the next steps that can be performed.

When fixed demands are considered, the best solution the constructive heuristic is able to obtain has an associated cost of 896.86, and it involves 11 routes. However, if demands are predicted taking into account the delivery order, the same heuristic obtains a solution with 8 routes and a cost of 791.26. Therefore, the savings might be noticeable when dynamic demands are considered.

Part II

APPLICATIONS

Chapter 6

Applications in transportation

This chapter studies several rich and realistic routing problems. It proposes different hybrid algorithms relying on metaheuristics, Monte Carlo simulation and regression models.
It is based on the following journal articles: Calvet et al. (submitted[a]), Calvet et al. (2016d), Gruler et al. (2016), Reyes et al. (submitted), and Calvet et al. (2016a). This work has been presented at the following conferences: Juan et al. (2015d), Juan et al. (2015b), Calvet et al. (2015a), and Calvet et al. (2015b).

6.1 Introduction

As a consequence of the growing flows of freight, the development of efficient and sustainable transportation and logistics activities has become a priority for Europe (European Union, 2011a; European Union, 2011b). The globalization, the growth of population, their purchase capacity and the efficiency of production systems are the main reasons of the increase in this sector. According to Eurostat (2015), emissions of greenhouses gases, air pollutants and noise from transport have significant impacts on the climate, the environment and human health. The need of flexible and efficient optimization tools affects both the public and the private sectors, since a huge number of companies daily address problems related to the transportation of people and/or goods.

In this context, the design of intelligent approaches is key for: *(i)* company competitiveness; *(ii)* good functioning of the labour market; *(iii)* cohesion within and between regions; *(iv)* reduction in the fossil energy importation, by the decrease of the consumption of petroleum derivatives; *(v)* decrement in the pollution, which improves people health; and *(vi)* reduction in traffic accidents.

The VRP is the most classical and simple formulation employed to describe logistic problems dealing with physical distribution. It constitutes the most popular research line in combinatorial optimization because of its practical relevance and its scientific interest due to its $\mathcal{NP}$-hardness. This chapter studies five realistic and rich extensions of the VRP (RVRP): the MDVRP-SD, the MDVRP-HD, the sustainable MDVRP, the WCP and the HSAVRP.

The MDVRP is a two-stage decision process, since assignation and routing issues are often interrelated, i.e., the assignment map may affect the quality of the posterior routing. Montoya-Torres et al. (2015) highlight a noticeable growing interest in the MDVRP during the last decade, with over 103 publications between 2006 and 2014. In the stochastic and capacitated MDVRP, a set of customers with random demands must be served by a fleet of homogeneous capacitated vehicles departing from one among several capacitated depots. The main goal is to determine the set of routes that minimizes the expected total routing cost, including recursive actions, subject to a number of capacity-related constraints. The problem has numerous applications in real-life, e.g.: garbage collection, gas distribution, stocking of vending machines, and other similar activities in which the specific amount of goods to leave or pick up is not known until the place is reached. Despite its relevance, there are few works on the stochastic MDVRP and, to the best of our knowledge, Calvet et al. (submitted[a]) is the first one addressing the stochastic and capacitated MDVRP.

Most works on the MDVRP has focused on minimizing distance-based distribution costs. However, no attention has been given so far to potential variations in demands due to the fitness of the customer-depot mapping in the case of heterogeneous depots. In the MDVRP-HD, the depots are heterogeneous in terms of their commercial offer, and customers show different willingness to consume depending on how well the assigned depot fits their preferences. As a consequence, market-segmentation strategies need to be considered to increase sales and total income while accounting for the distribution costs.

The increasing social concern is compelling companies to change purely commercial objectives in order to consider sustainability. This new vision seeks to compensate the negative impacts of transport activities without neglecting economic profits. In this context, it is essential to introduce approaches considering the impacts of the three pillars of sustainability for routing problems such as the MDVRP. For instance, travelling time and distance are related to economical impacts, carbon emissions to environmental impacts, and risk of accidents to social impacts.

In the face of rising population densities in urban areas around the world, a large number of cities are currently reorganizing their municipal responsibilities (Nations, 2015). As a consequence, the WCP is of high practical importance, especially in the context of smart city initiatives (Neirotti et al., 2014). On the one hand, uncollected garbage can lead to pollution of the environment and health-issues, while noise and road congestions through extensive use of waste collection vehicles decrease urban living standards. On the other hand, waste collection represents up to two thirds of operational waste management costs (Malakahmad et al., 2014; Son, 2014; Tavares et al., 2009). As waste generation and travel times of vehicles cannot be predicted with full certainty, there is a need for fast and risk-aware solutions of high quality which are able to take stochastic input variables into account.

The HSAVRP-SD considers a heterogeneous fleet. This diversity usually comes from two facts: different customers and locations may require different types of vehicle (e.g., due to narrow roads, available parking spaces, and vehicle weight restrictions), and the vehicle acquisitions may be made in different times and places. In addition, this problem describes a scenario where some customers cannot be accessed with all types of vehicle, which is known as site-dependency. Regarding the cost matrix, the classical assumption about its symmetry is relaxed, since there can be cost differences associated to the direction of a route (for instance, differences between driving uphill or downhill in mountainous regions). Moreover, the HSAVRP-SD also accounts for uncertainty in demands. It has several real-life applications such as the fuel oil distribution, which can be associated to petrol station replenishment or to the delivery of domestic heating oil. In these cases, the exact demand is not known until the time of the delivery, and cost between nodes (based on energy consumption) is asymmetric due to the presence of important road grades. The optimization of domestic heating oil distribution has been less studied, even though the high dependence on heating oil of some isolated regions. It could be of particular interest in mountainous regions where in absence of a gas pipeline, domestic oil is frequently the predominant fuel for heating.

6.2 Literature review

This section reviews relevance works related to the problems studied. First, it focuses on the MDVRP and the WCP. Afterwards, routing works considering stochasticity and sustainability are introduced. More references can be found in the articles introduced in this chapter.

6.2.1 The MDVRP

The MDVRP has been intensively studied in the last decades. Table 6.1 summarizes the information of the main related works.

6.2.2 The WCP

Probably the first work to address municipal solid waste collection is Beltrami and Bodin (1974). Since then, various solution techniques for different variants of the WCP have been proposed. While some works formulating the WCP as an ARP can be found (Ghiani et al., 2005; Bautista et al., 2008), the following discussion refers to recent publications using VRP formulations. More extensive literature reviews are provided by Beliën et al. (2014), Ghiani et al. (2014b), and Han and Ponce-Cueto (2015).

Most works on the WCP focus on case studies with some problem extension. For example, Baptista et al. (2002) elaborated an extension of the Christofides and Beasley heuristic for the multi-period WCP (Christofides and Beasley, 1984), modeled as a periodic VRP to combine vehicle scheduling over multiple time periods with route planning. Also addressing a multi-period WCP, Teixeira et al. (2004) developed a cluster-first route-second heuristic to schedule and plan

TABLE 6.1: Works on the MDVRP and extensions.

Work	Problem	Approach
Tillman and Cain (1972)	MDVRP	Branch and bound methods
Golden et al. (1977)	TSP, Multiple TSP, CVRP and MDVRP	Heuristics
Raft (1982)	MDVRP	Heuristic-based technique, which is decomposed into: route assignment, depot assignment, vehicle assignment, delivery period, and route design
Renaud et al. (1996)	Capacitated MDVRP with a maximum route length	TS
Cordeau et al. (1997)	Periodic VRP, periodic TSP and capacitated MDVRP	TS
Salhi and Sari (1997)	MDVRP determining the vehicle fleet composition	Three-level methodology, including composite heuristics
Thangiah and Salhi (2001)	MDVRP	GA, an insertion heuristic and a post-optimization method
Wu et al. (2002)	Capacitated multi-depot LRP with a heterogeneous fleet and a limited number of available vehicles	SA
Bae et al. (2007)	Capacitated MDVRP	GA and GUI-type programming
Crevier et al. (2007)	MDVRP with inter-depot routes	TS, a solution pool, a route generation algorithm, a set partitioning algorithm, and a post-optimization process
Pisinger and Ropke (2007)	CVRP, CVRP with time windows, capacitated MDVRP, site-dependent CVRP, and open CVRP	LNS
Chen and Xu (2008)	Capacitated MDVRP	GA and Metropolis acceptance rule of the SA
Ho et al. (2008)	MDVRP	Two hybrid GAs
Dondo and Cerdá (2009)	MDVRP with time windows	Mixed-integer linear programming formulations, a spatial decomposition scheme and a local search improvement algorithm exploring large neighborhoods
Mirabi et al. (2010)	Capacitated MDVRP	Three hybrid heuristics including nearest depot method, CWS heuristic, nearest neighbor heuristic, and SA
Gulczynski et al. (2011)	Multi-depot split delivery VRP	Integer programming-based heuristic
Surekha and Sumathi (2011)	Capacitated MDVRP	Clustering, CWS heuristic, and a GA
Cordeau and Maischberger (2012)	Capacitated MDVRP among other routing problems	ILS and TS
Vidal et al. (2012)	Capacitated MDVRP, periodic VRP, and capacitated multi-depot periodic VRP	Hybrid GA
Juan et al. (2015c)	Capacitated MDVRP	ILS and biased randomization
Escobar et al. (2014)	Capacitated MDVRP	Hybrid granular TS
Karakatic and Podgorelec (2015)	MDVRP	Survey of GAs
Li et al. (2015)	Capacitated MDVRP with simultaneous deliveries and pickups	ILS

waste collection routes for different waste types. Nuortio et al. (2006) presented a guided variable thresholding metaheuristic to solve a multi-period WCP. Hemmelmayr et al. (2013) addressed the periodic VRP with different waste types, which they solved with a VNS metaheuristic. The landfills are considered intermediate facilities, which are inserted in pre-constructed routes using dynamic programming. The authors also discussed the single period WCP with multiple depots, in which the landfills serve as vehicle depots and disposal sites at the same time. Later, Hemmelmayr et al. (2014) discussed the integrated vehicle routing- and bin allocation problem using the same real-life problem set, which they solved with a combination of a VNS metaheuristic for the routing part and a mixed integer linear programming-based exact method for the bin allocation. Ramos et al. (2014) extended the typical objective of minimizing routing costs in order to include environmental concerns, considering multiple waste types and numerous vehicle depots.

Only focusing on waste collection routing, Kim et al. (2006) developed an extension of Solomon's insertion algorithm (Solomon, 1987) to optimize routes of a waste management service provider, considering a capacitated vehicle fleet, time windows, and driver lunch breaks. A benchmark set of 10 realistic instances based on the original case study ranging from 102-2100 nodes is provided. This benchmark set was later employed by Ombuki-Berman et al. (2007) to test a multi-objective GA. Furthermore, the same benchmark set was used by Benjamin and Beasley (2010) and Buhrkal et al. (2012) to test their metaheuristic solution methods. Benjamin and Beasley (2010) combined the TS and the VNS metaheuristics. By exchanging containers and landfills within and between routes, the solution search space is systematically increased. Buhrkal et al. (2012) put forward an adaptive LNS metaheuristic. Based on an initial solution, this approach applies a range of destroy-and-repair methods to examine several solution neighborhoods. It is called adaptive since the choice of methods depends on the solution quality obtained during the construction of earlier solutions. Moreover, an acceptance criterion for new solutions based on the SA metaheuristic included. Recently, Markov et al. (2016) presented a multiple neighborhood search heuristic for a real-word application of the waste collection VRP with intermediate facilities. The authors consider a heterogeneous vehicle fleet and flexible depot destinations in their approach.

6.2.3 Stochasticity

Regarding the VRP-SD, the first works appear in the 80s. Table 6.2 shows some related works. It is worth highlighting a few outstanding contributions. In Dror and Trudeau (1986), the concept of route failure is introduced, and its effects on the expected cost of a route are illustrated. A review on the stochastic CVRP is presented in Gendreau et al. (1996), where the main variants are presented. Yang et al. (2000) suggest anticipating possible stock-outs by incorporating preventive breaks or restocking in the route design. The aim is to reduce the probability of route failure and, as a result, the cost. During the last decades, the scientific community has focused on the implementation of metaheuristics. In this context, Bianchi et al. (2006) compare the performance of several methodologies embedding one of the following metaheuristics: SA, TS, ILS, ACO, and EA.

The stochastic MDVRP

The number of works analyzing the stochastic MDVRP is rather limited. Tillman (1969) expands the CWS heuristic to address it. The procedure proposed may be applied to demands with Poisson, Exponential, Normal, Binomial or Chi-squared distributions. In Chan et al. (2001), a multi-depot, multiple-vehicle LRP with stochastically processed demands is formulated. The probable demands are estimated by stochastic processes before the vehicle location-routing decisions. Tatarakis and Minis (2009) study the stochastic MDVRP considering both the case in which products are stored dedicatedly or together in a compartment. Dynamic programming algorithms are proposed to determine the minimal routing cost, and an optimal routing policy is derived to decide whether a vehicle has to return to the depot for a reload after serving the current customer or should continue to the next customer. Tauhid et al. (2012) solve the stochastic MDVRP in three phases: first a nearest neighbor classification method is used for grouping the customers; then, the sum-of-subsets method is applied for routing; and finally, the routes are optimized throughout a greedy method. They aim to minimize the number of routes and, accordingly, the number of vehicles needed.

TABLE 6.2: Works on the VRP-SD and related problems.

Work	Problem	Approach	Modeling
Stewart and Golden (1983)	VRP-SD	Formulations and heuristics	Demands follow Normal or Poisson distributions
Dror and Trudeau (1986)	VRP-SD	Modification of the CWS heuristic	Demands follow specific distributions from a limited set
Bastian and Kan (1992)	VRP-SD	Formulation for a penalty model, a chance-constrained model, and a full-service model	Demands are independent and identically distributed with known probability distributions
Gendreau et al. (1995)	VRP-SD with stochastic customers	Stochastic integer formulation and integer L-shaped method	Demands are independent, discrete and identically distributed with known probability distributions
Yang et al. (2000)	VRP-SD	Two heuristics	Demands are independent and identically distributed with known probability distributions
Tan et al. (2007)	VRP-SD with time windows	MOEA and simulation	Demands follow Normal distributions
Ismail and Irhamah (2008)	VRP-SD	GA and TS	Demands follow independent and discrete Uniform distributions
Moghaddam et al. (2012)	CVRP with uncertain demands	PSO	No assumptions are made
Goodson (2015)	Multi-compartment VRP-SD	Method to calculate the expected cost of a solution integrated in a cyclic-order-based SA	Demands follow discrete distributions

None of the aforementioned works deals with the stochastic and capacitated MDVRP analyzed here. In particular, Tillman (1969) and Tauhid et al. (2012) consider unlimited capacities at each depot -which significantly reduces the difficulty of the problem and constitutes an unrealistic assumption. In addition, Tauhid et al. (2012) do not really consider stochastic demands. Also, Tillman (1969) makes strong assumptions on the probability distributions of these demands. Chan et al. (2001) and Tatarakis and Minis (2009) deal with problems that, although somewhat related, can not be considered stochastic and capacitated MDVRPs. While the former focuses on location issues, in the latter a single vehicle must deliver multiple products given a predefined customer sequence.

The stochastic WCP

Concerning the WCP with stochastic demands, the literature is more scarce. The ACO metaheuristic and a hybrid approach based on a GA and TS for a case study with 50 containers is presented in Ismail and Irhamah (2008), and Ismail and Loh (2009). After planning a priori routes, waste levels are simulated according to a discrete probability distribution. Routes undergo a recourse action (i.e., an additional disposal trip) whenever actual demand exceeds the planned collection amount. Nolz et al. (2014) formulated a collector-managed IRP for a case study on the collection of infectious waste. By using real information obtained through radio frequency identification, their ALNS algorithm is able to consider stochastic inputs. Alshraideh and Abu Qdais (2016) combined a multi-period WCP with time windows and stochastic demands. They used a GA and a probability constraint regarding a pre-defined service level to solve the problem.

6.2.4 Sustainability

The increasing social concern for the environment and a sustainable growth requires the transformation of cities. During the last decade, the green VRP (GVRP) and the pollution VRP (PVRP) have become increasingly popular. While the former is focused on the environmental impact caused by the fuel or energy consumption of transport, the latter takes into account the pollution and different emissions generated. Thus, both problems analyze the emissions and fuel/energy consumption levels, which depend on traffic congestion, speed, acceleration, type of road, type of vehicle, and load, among other internal and external factors of the operation (Bektaş and Laporte, 2011; Koç et al., 2014).

Regarding environmental impacts, the distance and vehicle weight play a crucial role in the fuel/energy consumption and carbon emissions, thereby Ubeda et al. (2011) aimed at reducing transport costs and emissions, considering the distance and some variations in the vehicle maximum capacity. It is concluded that enhancing load factors (which may be achieved by using heterogeneous fleets) is an efficient way to get significant savings and environmental benefits. The authors also discuss negative externalities of transport such as noise, air pollution, congestion, accident rate, energy consumption and land use, among others. There are studies tackling the negative impacts from three perspectives: negative externalities, emissions released and fuel consumption. Faulin et al. (2011), Liu et al. (2014), and Zhang et al. (2015) considered environmental indicators for the CVRP; they state that the load variation defines fuel consumption and emissions caused by transport. Besides, the load variation influences the distribution processes profitability. In this line, Kuo (2010), Demir et al. (2014), and Xiao and Konak (2015) developed methodologies for the green heterogeneous VRP, considering traffic congestion, road gradient, speed variations, and distance traveled as variables that influence fuel consumption and as elements that characterize the urban transport dynamics (Jabbarpour et al., 2015). More recently, Niknamfar and Niaki (2016) study the MDVRP with time windows to optimize the customers-depots allocation and the vehicles selection aiming to minimize the environmental impacts. They demonstrated that an optimal allocation and coordination between stakeholders not only reduce the negative impacts but also enhance the total profit. Juan et al. (2014c) considered a supply chain with multiple suppliers for minimizing the empty trips and the travel distance in each route. They concluded that it is possible to reduce the CO2 emission to 23% when the distribution process is carried out in collaboration with multiple suppliers. Wang et al. (2014b) demonstrated that considering environmental criteria allows a saving up to 10% of the operation costs. The authors developed an algorithm to integrate the economic and environmental goals based on the MDVRP with backhauls. Demir

et al. (2014) considered the MDVRP with freight pick-up and delivery to ensure that any customer demand can be met from any depot and thus reducing the operation cost.

Some studies have focused on the analysis of environmental impacts caused by transport activities in urban zones, however there is no characterization for getting a rough estimation of the real impact of these activities. For instance, about 60% of transport activities take place in urban regions at where around 80% population is concentrated, making people the main harmed (European Commission, 2015). Social impact refers to health problems and other factors such as quietness, air quality, urban esthetic, accessibility and urban safety. Penalties, taxes or willingness to pay as a means to reduce the social impacts constitute the costs associated. It is estimated that about 0.4%, 0.2%, 1.5% and 2% of the gross domestic product is related to air pollution problems, noise, accidents and traffic congestion, respectively (Caceres et al., 2014). Therefore, the sustainability concept has started to take part in the decision-making process but there is a lack of structured tools that allow the integration of the three dimensions and support decision-makers (Chen et al., 2013).

There are only a few works on sustainability criteria. Chibeles-Martins et al. (2016) pose ecological criteria to determine an optimal structure of distribution networks. They solved a bi-objective problem focused on determining the suitable locations, capacities and attributes in factories, warehouses and a distribution center. The solution method is based on the SA metaheuristic and Pareto optimality is considered to get a balancing between economic and ecological concerns. In the same sense, Zhang et al. (2016) implement EAs to determine the optimal design of supply chains considering two possible scenarios: first, the transport is outsourced and second the transport is leased. It is a multi-objective problem aimed at minimizing CO2 emissions, fine dust and costs. The authors implement the non-dominated sorting GA-II (NSGA-II) and the strength Pareto EA2 (SEAP2) to compare their performance, both methods take into account Pareto optimality through a scalarization method computed by a weighted sum. Later, Kadziński et al. (2017) define a sustainable objective to design an optimal distribution structure considering a supply chain with multi-distribution channels. Objectives are maximizing customer coverage, and minimizing cost and environmental impacts. Notice that social objectives do not respond to problems highlighted by the society, besides these approaches belong to strategic levels without considering the synergy among tactical levels, operative levels and stakeholders' particular objectives.

6.3 The MDVRP-SD

The stochastic and capacitated MDVRP is characterized by the randomness of at least one of its parameters or structural variables. These random variables follow specific probability distributions. This problem may be seen as a non-trivial extension of the stochastic CVRP (Gendreau et al., 1996; Stewart and Golden, 1983). There are three problems belonging to this family: the CVRP with stochastic demands, which is the most popular (Bianchi et al., 2006); the CVRP with stochastic customers (Bertsimas, 1988); and the CVRP with stochastic times (Laporte et al., 1992; Kenyon and Morton, 2003). The MDVRP may be described as follows. Let $G = \{V, E\}$ be a complete directed graph, where $V = \{V_d, V_c\}$ is the set of vertices including the depots (V_d) and the customers (V_c), and E is the set of edges connecting all vertices in V. Each customer $i \in V_c$ has a positive demand d_i. Each depot $p \in V_d$ has assigned a maximum number of vehicles, m. All vehicles are supposed to have the same capacity W. Each edge in E has an associated cost $c_{ij} = c_{ji} \geq 0$. A solution is a set of routes in which each route starts at one depot in V_d, connects one or more customers in V_c, and ends at the same depot (Figure 6.1). Moreover, each customer must be visited only once. The MDVRP-SD differs in the following two consideration: *(i)* each customer has a positive demand D_i that follows a probability distribution, either theoretical or empirical, with an existing mean denoted as $E[D_i]$; and *(ii)* each customer is visited once except in the undesirable case in which a route failure occurs. While the demands' distribution is known beforehand, the exact demand cannot be revealed until the vehicle reaches the customer.

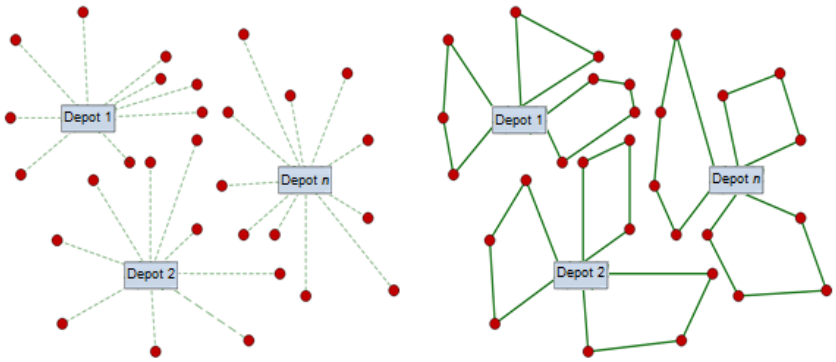


FIGURE 6.1: Customer-depot assignment and posterior routing processes in the MDVRP.

The classical goal of the MDVRP-SD is to find a solution that minimizes the expected routing cost while satisfying the customer demands, and the constraints related to the number of vehicles, and the vehicles' capacity. However, other constraints may also apply, e.g.: a maximum allowable cost for a route, time windows for visiting each customer, etc. In addition, different goals may be proposed such as solution balance or minimization of environmental costs. Even in its simplest version, this problem represents a challenge since it integrates a combinatorial assignment problem, in which each customer is assigned to one depot, with several stochastic CVRPs, one per depot. The additional complexity lies in the interrelation between assigning and routing issues.

One way to model the MDVRP-SD is as a two-stage problem. In the first stage (design stage), a set of routes is designed considering the probability distributions associated with each customer's demand. The second stage (corrective stage) specifies the actual route of each vehicle, which may include corrective actions if the route fails, i.e., if the demand of a customer visited by a given vehicle is higher than the remaining vehicle capacity. In this case, the vehicle must return to the depot to reload. Often, the possibility of re-stocking is allowed, that means that a vehicle may return to the depot before it has run out of capacity. For instance, if the remaining vehicle capacity is not enough to satisfy the expected demands of the customers that still have to be served. The solution must minimize the expected total cost, which is the sum of the costs of the routes planned in the first stage (fixed cost), and the expected costs due to corrective actions (variable cost).

6.3.1 Methodology

Our approach relies on two facts: *(i)* the MDVRP-SD can be considered a generalization of the MDVRP, i.e., the MDVRP can be seen as a MDVRP-SD in which the random demands have zero variance; and *(ii)* despite the fact that the MDVRP-SD has not been intensively studied, there exists efficient algorithms for solving the MDVRP.

The general ideas behind our approach are described next. Initially, given an instance of the MDVRP-SD, it is transformed into a deterministic instance by replacing each random variable by its mean. A set of high-quality solutions for the deterministic version is then obtained by using an efficient algorithm. While the search takes place, MCS techniques are employed to assess the performance of these promising solutions for the stochastic version. The best solution is defined as the one with the lowest expected total cost. The proposed methodology employs safety stocks as suggested in Juan et al. (2011b). A safety stock is a certain amount of the vehicle capacity that is not considered while designing the routes. Then, if the final routes' demands surpass their expected values, this stock can be employed to try to satisfy them. Thus, the aim of considering safety stocks is to reduce the probability of a route failure.

Proposed steps

The flowchart diagram is depicted in Figure 6.2 and described next:

1. Consider a MDVRP-SD instance defined by a set of n customers. Each customer i has associated a demand D_i ($1 \leq i \leq n$) that follows a known probability distribution with an existing mean $E[D_i]$.

2. Determine a set K of percentages, where each element k_l is the percentage of the vehicle capacity (W) that can be used during the route design phase; in other words, $1-k_l$ represents a fixed level of safety stock. For each of these elements, follow the steps 3 to 9.

3. Consider the capacitated MDVRP(k_l) with a total vehicle capacity of $W_l^* = k_l \cdot W$ and deterministic demands $d_i = E[D_i]$.

4. Generate an initial solution for the MDVRP(k_l). This solution is also an aprioristic solution for the MDVRP-SD. It will be employed "as it is" as long as there is no need of corrective actions (routes failures and re-stockings). Therefore, the cost associated to this solution, $C_{MDVRP}(k_l)$, can be considered a base or fixed cost of the MDVRP-SD solution. In the case of the stochastic problem, there is also a variable cost $C_{CA}(k_l)$ that depends on the corrective actions undertaken. Consequently, for a given value of k_l, the total cost of the 'stochastic' solution (the one associated with the MDVRP-SD) is the sum of the fixed cost corresponding to the 'deterministic' solution (the one associated with the MDVRP) and the variable cost due to corrective actions, $C_{MDVRP-SD}(k_l) = C_{MDVRP}(k_l) + C_{CA}(k_l)$.

5. Use MCS to estimate the expected cost due to corrective actions for each route j of the aprioristic solution, $E[C_{CA}^j(k_l)]$ $(1 \leq j \leq m)$. Then, aggregate the expected total cost for all routes, $E[C_{CA}(k_l)]) = \sum_{j=1}^{m} E[C_{CA}^j(k_l)]$. In this phase, a short simulation is used to quickly get that estimate. Then, the expected total cost of the solution is calculated as follows: $E[C_{MDVRP-SD}(k_l)] = C_{MDVRP}(k_l) + E[C_{CA}(k_l)]$.

6. Set a base solution as the initial solution.

7. Employ a metaheuristic algorithm, which starts an improvement process that will continue until a stopping condition, based on time or a fixed number of iterations, is reached. At each iteration the following steps are implemented. First, a perturbation is applied to the base solution to generate a new one. If the fixed cost of the new solution is lower than the fixed cost of the current base solution, then the list of the best deterministic solutions is updated (only if it is not full or if the worst solution has a higher cost, then a swap is performed) and the expected total cost of the new solution is estimated with a short simulation. If this cost is lower than the expected total cost of the base solution, the latter is replaced and the list of the best stochastic solutions is updated. Otherwise, an acceptance criterion is used to decide whether the base solution is deteriorated to the new one. Before that, if the fixed cost of the new solution is higher, then that solution is discarded. This iterative process will provide, after analyzing many possible solutions, a list of promising solutions for the MDVRP-SD.

8. Try to improve all promising solutions with an intensive routing algorithm.

9. Use a long simulation to generate a sample of total costs for each promising solution. Large samples are required to obtain estimates with small confidence intervals.

10. Finally, return the top best stochastic solutions (considering all solutions found with the different values in K), and the corresponding samples (they will be used for completing a risk analysis).

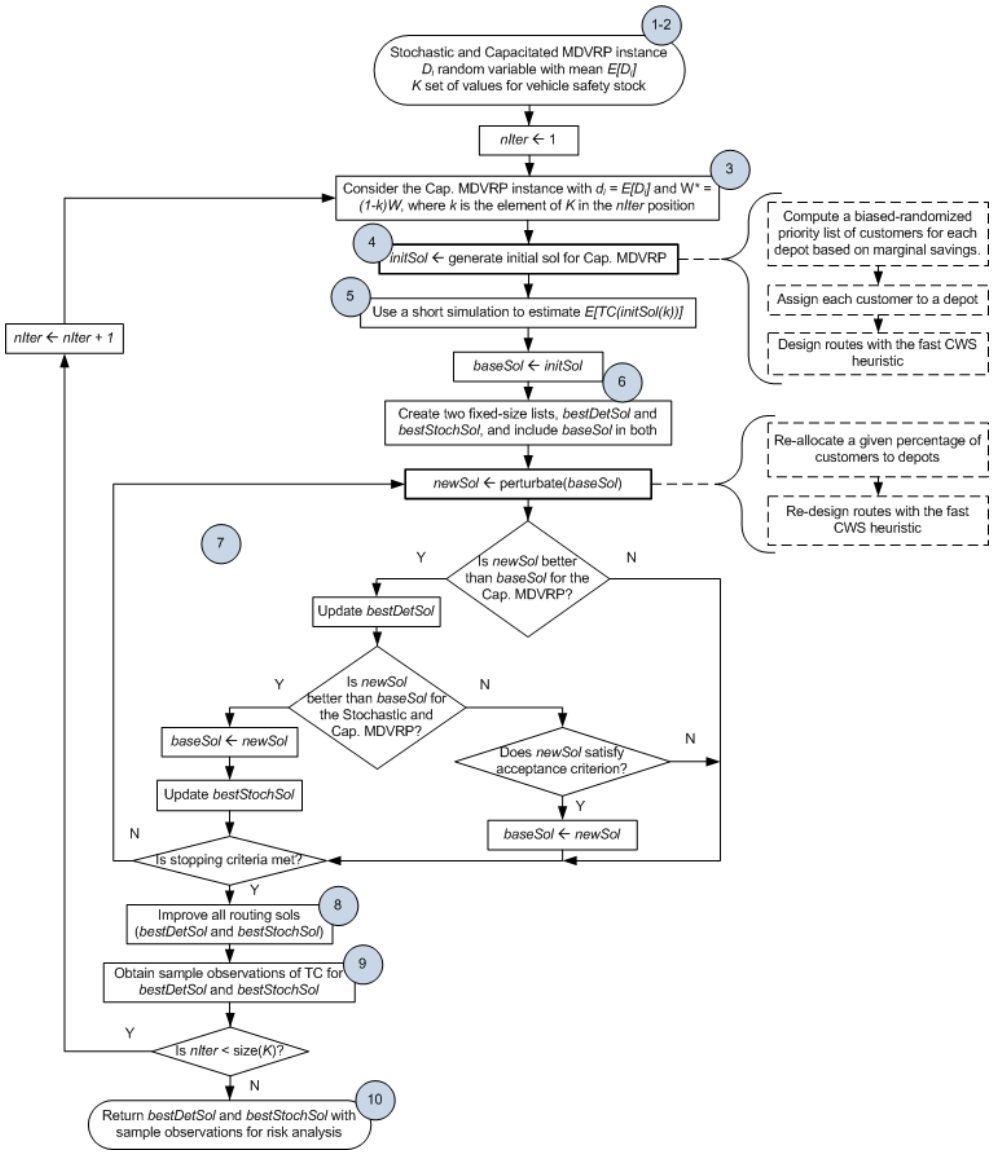


FIGURE 6.2: Flowchart of the proposed approach for the MDVRP-SD.

Details and further considerations

The algorithm used to get the initial solution for the MDVRP is the one proposed in Juan et al. (2015c). Initially, a customer-depot assignment map is set, and then the savings heuristic (Clarke and Wright, 1964) is applied to obtain a fast routing plan. Regarding safety stocks, it is expected that lower values of k_l will provide more reliable routes, as a high percentage of the vehicle capacity will be reserved as safety stock. However, a high fixed cost will result too, since more vehicles will be needed to cover all the customers' demands. On the other hand, a high value of k_l is related to a lower fixed cost but a higher variable cost due to the elevated risk of having to return to the depot to reload. Considering the trade-off between these two costs, we try different values as indicated in step 2.

The cost due to corrective actions is computed as follows. In case of route failure, it includes the cost of returning to the depot first and then to the customer being served. It is assumed that the vehicle delivers all the remaining stock before going back to reload. A re-stocking is carried out when the expected demand of the next customer is higher than the current remaining stock. The cost of this strategy incorporates the costs on the edges that link a customer with the depot and the depot with the next customer minus the cost of the edge linking both customers.

The perturbation operator modifies the current solution by reallocating a given percentage p of customers randomly selected, considering the remaining capacity of the depots, and the distance-based cost for each pair of customer and depot. The savings heuristic is applied again to design the routes. A Demon-like acceptance criterion (Talbi, 2009) is used to diversify the search.

In order to improve the most promising solutions found within the metaheuristic framework, the routing algorithm proposed by Juan et al. (2011a) is applied to each one. This algorithm is based on a randomized version of the savings heuristic that employs a Geometric distribution to guide the random search, and a cache and splitting techniques to make it more efficient. This algorithm has been adapted for the stochastic solutions.

Finally, it is interesting to observe that, considering the parameters (not the estimates), the fixed cost and the expected total cost of the best deterministic solution represent a lower and an upper bound, respectively, of the expected total cost associated to the best stochastic solution. The set of samples will allow us to compare the solutions not only focusing on the expected total cost, but also on the distribution of the total cost.

6.3.2 Computational experiments

The algorithm has been implemented as a Java application and tested on 23 MDVRP benchmark instances: the first seven were proposed by Christofides and Eilon (1969), the following four were created by Gillett and Johnson (1976) and the remaining are described in Chao et al. (1993). Vidal et al. (2012), Escobar et al. (2014), and Juan et al. (2015c) are some recent works using them. These instances have been adapted as described next. The demand of each customer (d_i) has been considered as a random variable D_i following a Lognormal distribution with mean d_i and variance vd_i. Three different scenarios have been considered, each one with a respectively different variance: 0.1 $E[D_i]$, 0.5 $E[D_i]$, and 1 $E[D_i]$, where $E[\cdot]$ represents the mean or expected value. In order to choose the percentage of the vehicle capacity in the route design phase ($1 - k_l$), 5 equally-spaced values varying from 0.90 to 1.00 have been tested.

The computational time is limited to 30 seconds. The number of seeds is set to 10, and only the best result are stored. Concerning the number of iterations in each simulation, 200 runs have been employed for the short simulations and 2,000 runs for the long simulations. The selection of these values, as well as of the number of solutions stored in the list of top solutions (4), is mainly driven by the total computing time available. Biased randomization techniques rely on two Geometric distributions (one for mapping and one for routing) and, therefore, they require distribution parameters: bM and bR, respectively. Additionally, there is a parameter p which controls the percentage of nodes that may be reallocated in a solution when perturbing it. They have been tuned by performing a full factorial experiment. bM, bR and p follow Uniform distribution between $[0.3, 0.4]$, $[0.2, 0.3]$ and $[0.3, 0.4]$.

Results are displayed in Tables 6.3, 6.4, and 6.5. Each of these tables represents a specific scenario. The first column identifies the instance and the second shows the best known solution (BKS) for the MDVRP. The next five columns are associated with the solution with the lowest fixed cost: the first, the *best deterministic solution - fixed cost* (BDS-FC), represents the fixed cost; the second calculates the gap between the BKS and the BDS-FC, which reveals the performance of our algorithm for the deterministic version of the problem; the third, the *best deterministic solution - total expected cost* (BDS-TEC), is the expected total cost; the fourth, the *best deterministic solution - reliability* (BDS-R), has been computed as one minus the number of route failures divided by the number of routes; and the fifth, *best deterministic solution - k* (BDS-K), provides the percentage of the vehicle total capacity chosen. The following column represents the gap between the BDS-TEC and the BDS-FC. The next three columns are associated with the solution with the lowest expected total cost: the *best stochastic solution - total expected cost* (BSS-TEC) contains the expected total cost; the following two columns, the *best stochastic solution - reliability* (BSS-R) and the *best stochastic solution - k* (BSS-K), show the associated reliability, and the k-value respectively. The next two columns are the gaps between the BSS-TEC and the BKS, and between the BSS-TEC and the BDS-FC, respectively. It is important to highlight that, provided a 'large' number of simulation iterations is used, the BSS-TEC is bounded by the BDS-FC and the BKS (lower bounds), and the BDS-TEC (upper bound). Therefore, the previous gaps show the difference between the BSS-TEC and its lower bounds. The last column is the gap between the expected total costs of both solutions.

TABLE 6.3: Table of results for the MDVRP benchmark instances with a low demand variability.

Inst.	BKS (1)	BDS-FC (2)	G. (2)-(1)	BDS-TEC (3)	BDS-R	BDS-K	G. (3)-(2)	BSS-TEC (4)	BSS-R	BSS-K	G. (4)-(1)	G. (4)-(2)	G. (4)-(3)
p01	576.87	583.11	1.08%	594.85	1.00	1.00	2.01%	594.85	1.00	1.00	3.12%	2.01%	0.00%
p02	473.53	480.72	1.52%	487.43	1.00	1.00	1.39%	483.63	1.00	0.925	2.13%	0.60%	-0.78%
p03	641.19	648.60	1.16%	668.71	1.00	1.00	3.10%	654.34	1.00	0.95	2.05%	0.88%	-2.15%
p04	1001.04	1043.05	4.20%	1108.63	1.00	1.00	6.29%	1073.77	1.00	0.95	7.27%	2.95%	-3.14%
p05	750.03	775.13	3.35%	792.15	1.00	1.00	2.19%	778.85	1.00	0.975	3.84%	0.48%	-1.68%
p06	876.50	897.16	2.36%	966.85	1.00	1.00	7.77%	924.28	1.00	0.925	5.45%	3.02%	-4.40%
p07	881.97	899.96	2.04%	961.25	1.00	1.00	6.81%	940.16	1.00	0.975	6.60%	4.47%	-2.19%
p08	4371.66	4495.17	2.83%	5058.74	0.99	1.00	12.54%	4623.25	1.00	0.975	5.75%	2.85%	-8.61%
p09	3858.66	3962.64	2.69%	4273.86	1.00	1.00	7.85%	4056.49	1.00	0.975	5.13%	2.37%	-5.09%
p10	3629.60	3738.80	3.01%	3925.71	1.00	1.00	5.00%	3794.06	1.00	0.975	4.53%	1.48%	-3.35%
p11	3545.18	3612.26	1.89%	3815.84	1.00	1.00	5.64%	3677.32	1.00	0.975	3.73%	1.80%	-3.63%
p12	1318.95	1318.95	0.00%	1326.89	1.00	1.00	0.60%	1326.74	1.00	0.975	0.59%	0.59%	-0.01%
p13	1318.95	1318.95	0.00%	1326.71	1.00	0.95	0.59%	1326.68	1.00	1.00	0.59%	0.59%	0.00%
p14	1360.12	1360.12	0.00%	1361.93	1.00	0.90	0.13%	1361.50	1.00	0.975	0.10%	0.10%	-0.03%
p15	2505.42	2557.53	2.08%	2666.42	1.00	1.00	4.26%	2590.15	1.00	0.925	3.38%	1.28%	-2.86%
p16	2572.23	2587.86	0.61%	2616.13	1.00	0.975	1.09%	2616.13	1.00	0.975	1.71%	1.09%	0.00%
p17	2709.09	2714.66	0.21%	2718.27	1.00	1.00	0.13%	2718.27	1.00	1.00	0.34%	0.13%	0.00%
p18	3702.85	3812.30	2.96%	3841.51	1.00	1.00	0.77%	3833.52	1.00	0.975	3.53%	0.56%	-0.21%
p19	3827.06	3876.15	1.28%	3918.52	1.00	1.00	1.09%	3918.52	1.00	1.00	2.39%	1.09%	0.00%
p20	4058.07	4085.91	0.69%	4091.26	1.00	0.90	0.13%	4091.26	1.00	0.90	0.82%	0.13%	0.00%
p21	5474.84	5681.16	3.77%	5849.68	1.00	1.00	2.97%	5741.34	1.00	0.925	4.87%	1.06%	-1.85%
p22	5702.16	5808.74	1.87%	5864.13	1.00	0.975	0.95%	5864.13	1.00	0.975	2.84%	0.95%	0.00%
p23	6078.75	6140.01	1.01%	6147.81	1.00	0.90	0.13%	6147.47	1.00	0.90	1.13%	0.12%	-0.01%

TABLE 6.4: Table of results for the MDVRP benchmark instances with a medium demand variability.

Inst.	BKS (1)	BDS-FC (2)	G. (2)-(1)	BDS-TEC (3)	BDS-R	BDS-K	G. (3)-(2)	BSS-TEC (4)	BSS-R	BSS-K	G. (4)-(1)	G. (4)-(2)	G. (4)-(3)
p01	576.87	580.49	0.63%	621.78	0.99	1.00	7.11%	607.92	1.00	0.925	5.38%	4.73%	-2.23%
p02	473.53	481.05	1.59%	495.72	0.99	1.00	3.05%	484.47	1.00	0.925	2.31%	0.71%	-2.27%
p03	641.19	648.80	1.19%	672.36	1.00	1.00	3.63%	663.64	1.00	0.925	3.50%	2.29%	-1.30%
p04	1001.04	1039.64	3.86%	1194.31	0.98	1.00	14.88%	1136.68	0.99	0.95	13.55%	9.33%	-4.83%
p05	750.03	775.40	3.38%	814.19	0.99	1.00	5.00%	797.13	1.00	0.95	6.28%	2.80%	-2.10%
p06	876.50	896.83	2.32%	991.64	0.98	1.00	10.57%	947.47	1.00	0.925	8.10%	5.65%	-4.45%
p07	881.97	901.93	2.26%	988.31	0.98	1.00	9.58%	975.88	0.99	0.975	10.65%	8.20%	-1.26%
p08	4371.66	4496.87	2.86%	5173.76	0.99	1.00	15.05%	4726.09	1.00	0.95	8.11%	5.10%	-8.65%
p09	3858.66	3981.77	3.19%	4290.58	0.99	1.00	7.76%	4116.24	1.00	0.95	6.68%	3.38%	-4.06%
p10	3629.60	3754.90	3.45%	4032.78	1.00	1.00	7.40%	3866.11	1.00	0.95	6.52%	2.96%	-4.13%
p11	3545.18	3607.04	1.75%	3953.10	0.99	1.00	9.59%	3767.31	1.00	0.925	6.27%	4.44%	-4.70%
p12	1318.95	1318.95	0.00%	1356.83	1.00	0.975	2.87%	1356.83	1.00	0.975	2.87%	2.87%	0.00%
p13	1318.95	1318.95	0.00%	1355.84	1.00	0.975	2.80%	1355.84	1.00	0.975	2.80%	2.80%	0.00%
p14	1360.12	1360.12	0.00%	1393.37	1.00	0.90	2.45%	1392.10	1.00	0.925	2.35%	2.35%	-0.09%
p15	2505.42	2549.17	1.75%	2722.47	0.99	1.00	6.80%	2635.44	1.00	0.90	5.19%	3.38%	-3.20%
p16	2572.23	2587.86	0.61%	2671.31	1.00	0.975	3.22%	2671.31	1.00	0.975	3.85%	3.22%	0.00%
p17	2709.09	2714.66	0.21%	2783.04	1.00	1.00	2.52%	2783.04	1.00	1.00	2.73%	2.52%	0.00%
p18	3702.85	3814.73	3.02%	3930.95	1.00	1.00	3.05%	3916.54	1.00	0.925	5.77%	2.67%	-0.37%
p19	3827.06	3875.40	1.26%	4002.83	1.00	1.00	3.29%	4002.83	1.00	1.00	4.59%	3.29%	0.00%
p20	4058.07	4085.91	0.69%	4187.05	1.00	0.90	2.48%	4187.05	1.00	0.90	3.18%	2.48%	0.00%
p21	5474.84	5690.22	3.93%	5891.98	1.00	1.00	3.55%	5870.96	1.00	0.925	7.24%	3.18%	-0.36%
p22	5702.16	5796.48	1.65%	5980.72	1.00	0.975	3.18%	5980.72	1.00	0.975	4.89%	3.18%	0.00%
p23	6078.75	6140.01	1.01%	6290.98	1.00	0.90	2.46%	6289.53	1.00	0.90	3.47%	2.44%	-0.02%

TABLE 6.5: Table of results for the MDVRP benchmark instances with a high demand variability.

Inst.	BKS (1)	BDS-FC (2)	G. (2)-(1)	BDS-TEC (3)	BDS-R	BDS-K	G. (3)-(2)	BSS-TEC (4)	BSS-R	BSS-K	G. (4)-(1)	G. (4)-(2)	G. (4)-(3)
p01	576.87	587.18	1.79%	633.62	0.99	1.00	7.91%	620.93	1.00	0.925	7.64%	5.75%	-2.00%
p02	473.53	479.45	1.25%	491.96	0.99	1.00	2.61%	485.77	1.00	0.90	2.58%	1.32%	-1.26%
p03	641.19	649.87	1.35%	679.88	1.00	1.00	4.62%	671.69	1.00	0.925	4.76%	3.36%	-1.20%
p04	1001.04	1042.05	4.10%	1197.08	0.97	1.00	14.88%	1177.35	0.98	0.95	17.61%	12.98%	-1.65%
p05	750.03	777.41	3.65%	821.53	0.98	1.00	5.67%	806.97	0.99	0.95	7.59%	3.80%	-1.77%
p06	876.50	897.48	2.39%	1021.06	0.97	1.00	13.77%	971.84	0.99	0.925	10.88%	8.29%	-4.82%
p07	881.97	907.38	2.88%	1011.02	0.97	1.00	11.42%	991.80	0.97	1.00	12.45%	9.30%	-1.90%
p08	4371.66	4498.65	2.90%	5267.60	0.98	1.00	17.09%	4772.74	1.00	0.925	9.17%	6.09%	-9.39%
p09	3858.66	3962.30	2.69%	4398.51	0.99	1.00	11.01%	4185.01	1.00	0.95	8.46%	5.62%	-4.85%
p10	3629.60	3747.91	3.26%	4106.36	0.99	1.00	9.56%	3940.02	1.00	0.95	8.55%	5.13%	-4.05%
p11	3545.18	3625.26	2.26%	4010.29	0.99	1.00	10.62%	3788.72	1.00	0.925	6.87%	4.51%	-5.53%
p12	1318.95	1318.95	0.00%	1377.66	1.00	1.00	4.45%	1377.66	1.00	1.00	4.45%	4.45%	0.00%
p13	1318.95	1318.95	0.00%	1376.28	0.99	1.00	4.35%	1376.28	0.99	1.00	4.35%	4.35%	0.00%
p14	1360.12	1360.12	0.00%	1417.01	0.99	0.90	4.18%	1414.06	0.99	0.925	3.97%	3.97%	-0.21%
p15	2505.42	2553.90	1.93%	2755.74	0.98	1.00	7.90%	2673.11	1.00	0.90	6.69%	4.67%	-3.00%
p16	2572.23	2590.77	0.72%	2712.30	0.99	0.975	4.69%	2712.30	0.99	0.975	5.45%	4.69%	0.00%
p17	2709.09	2714.66	0.21%	2823.97	1.00	0.90	4.03%	2823.97	1.00	0.90	4.24%	4.03%	0.00%
p18	3702.85	3813.22	2.98%	4005.61	0.99	0.975	5.05%	3971.55	1.00	0.925	7.26%	4.15%	-0.85%
p19	3827.06	3876.15	1.28%	4054.76	0.99	1.00	4.61%	4054.76	0.99	1.00	5.95%	4.61%	0.00%
p20	4058.07	4085.91	0.69%	4252.71	0.99	0.90	4.08%	4252.71	0.99	0.90	4.80%	4.08%	0.00%
p21	5474.84	5677.62	3.70%	5974.19	0.99	1.00	5.22%	5957.66	0.99	0.925	8.82%	4.93%	-0.28%
p22	5702.16	5812.03	1.93%	6079.27	0.99	0.975	4.60%	6079.27	0.99	0.975	6.61%	4.60%	0.00%
p23	6078.75	6140.01	1.01%	6388.07	1.00	0.90	4.04%	6386.58	1.00	0.975	5.06%	4.02%	-0.02%

TABLE 6.6: Summary of results for the MDVRP benchmark instances.

Scenario	G. BDS-FC - BKS	G. BDS-TEC - BDS-FC	G. BSS-TEC - BKS	G. BSS-TEC - BDS-FC	G. BSS-TEC - BDS-TEC
Var: $0.10E[D_i]$	1.83%	3.10%	3.12%	1.26%	−1.69%
Var: $0.50E[D_i]$	1.83%	5.89%	5.53%	3.62%	−2.06%
Var: $1.00E[D_i]$	1.74%	7.48%	7.12%	5.27%	−1.97%

6.3.3 Analysis of results

The results obtained show that assuming a problem being deterministic can lead to solutions with poor performance even in scenarios characterized by demands with a relatively low variance. In all experiments, the expected total cost obtained with the best stochastic solution is better than the one obtained with the best deterministic solution. There is a case in which the gap reaches the −9.39% (instance p08 with high variance). The reason is that the deterministic solution is not balanced, and a high variance results in an increasing of the expected total cost. Figure 6.3 illustrates the case of the instance p02 with a high variance. The vehicle capacity is 160. The left and right plots represent the best deterministic solution and the best stochastic solution, respectively. The numbers in the nodes reveal the expected customer demands, while the numbers in the center of each route are the total demands. Although the routes are similar, notice that the best stochastic solution seems more 'balanced' in terms of demands, which explains why it is also more reliable.

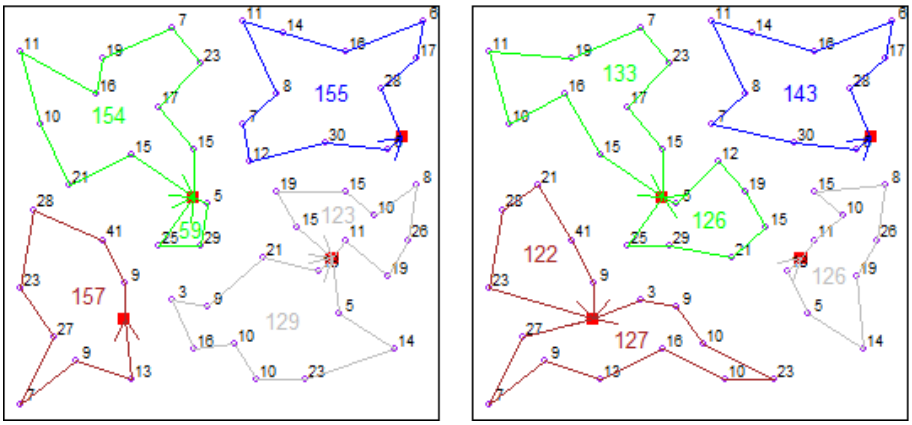


FIGURE 6.3: Best deterministic (left) and stochastic solutions (right) for the MDVRP instance p02.

Table 6.6 summarizes the results described in Tables 6.3, 6.4 and 6.5. For each scenario, it shows the mean gaps. The means gaps between the BDS-FC and the BKS, which ranges from 1.74% to 1.83%, show that our approach is relatively competitive for finding the best solution to the deterministic problem. The third column reveals that the difference between the BDS-TEC and the BDS-FC (i.e., the total cost if there was no stochasticity) is positive and positively correlated with the variability of the scenario. Next two columns quantify the gaps between the BSS-TEC and its lower bounds, the BKS and the BDS-FC. They both increase as the variability of the scenario gets higher. Finally, the last gap shows the benefit of using the simheuristic approach. Thus, it can be concluded that the higher the variability the higher the benefit.

Here we present a risk analysis in which the four best stochastic solutions and the best deterministic solution are compared. It is illustrated on a specific case, the instance p09 with high variance. Thus, Figure 6.4 shows a boxplot of the total costs obtained by means of MCS. It can be stated that the variability of total costs associated to the best deterministic solution is the highest, and all distributions present a positive skew. In Figure 6.5, the empirical cumulative distributions functions (CDFs) for the best deterministic and stochastic solutions are drawn. The probability distribution function of the best stochastic solution is above the other almost for the entire domain. In other words, the probability of having a total cost equal to or lower than a given value is usually higher with this solution. As a consequence, a risk-averse decision-maker would prefer it. Nevertheless, the minimum values are provided by the deterministic solution, which makes sense since

this solution will be the one selected in scenarios where the customer demands are similar to the corresponding mean of the distributions.

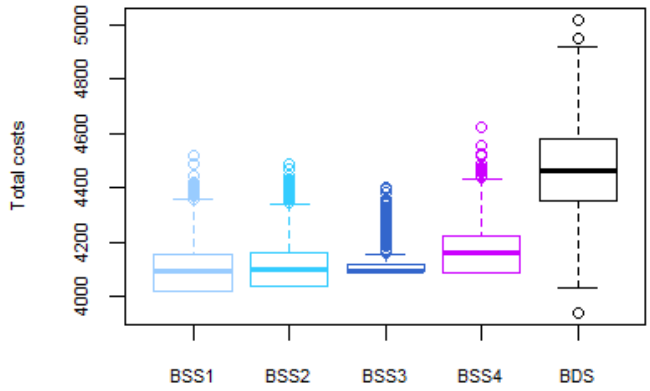


FIGURE 6.4: Boxplots of best solutions for the MDVRP instance p09 with high variability.

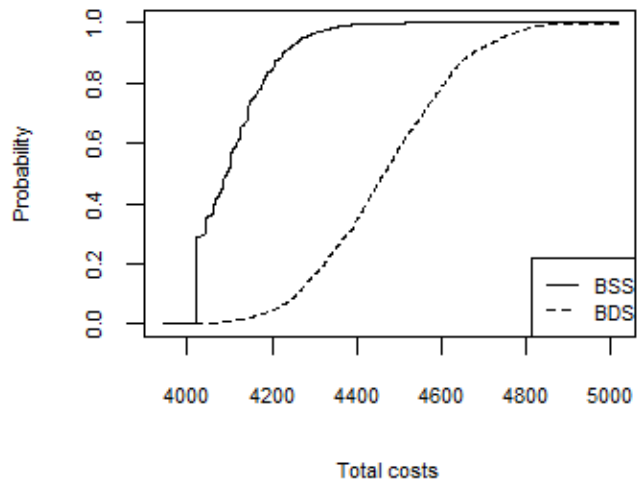


FIGURE 6.5: CDFs of best deterministic and stochastic solutions for the MDVRP instance p09 with high variability.

6.4 The MDVRP-HD

The problem addressed here is an extension of the MDVRP, already introduced in this chapter. When adopting a marketing perspective, companies focus on market segmentation to group customers according to their features and preferences. Considering the heterogeneity of markets,

segmentation attempts to divide customers into subsets that behave in a similar way. This extension of the MDVRP aims at assigning customers to depots based not only on distribution costs but also on customers' features and preferences. The goal is to optimize expected benefits by considering both distribution costs and expected incomes.

6.4.1 Methodology

In order to assign customers to depots, the heterogeneity of the depots is taken into account. It is a realistic approach, since depots belonging to the same organization usually have different characteristics related to products, trade credit policies, and complementary services, among others. This diversity leads to consider customer preferences. Specifically, the willingness to consume (or expenditure) of each customer depends on how well the assigned depot fits his/her preferences. Market segmentation techniques are applied to identify subsets of customers with similar profiles and assign them to the particular depot that better fits their preferences, considering the restrictions of the problem. Accordingly, it is proposed to study the relationship between expenditure and customers' features from data of existent customers by employing statistical learning methodologies (e.g., prediction techniques). It will enable the assignation of new customers in such a way that the expected benefits (expected incomes minus distribution costs) is maximized. The phases of the proposed approach are represented in Figure 6.6 and described next:

1. Data collection. The approach requires several inputs: database of historical sales, description of new customers, location of depots, vehicle maximum capacity, number of available vehicles at each depot, and maximum distribution costs per route. The sales database includes the following information for each existent customer: personal features, geographical location, expenditure level, and depot to which he/she has been assigned (randomly or according to a metric not related to personal features such as distribution costs). The description of new customers gathers personal features and geographical locations. Regarding the information of both existent and new customers, an initial selection of variables has to be performed by assessing which ones may be valuable. Besides explaining the differences of expenditures among depots, they should be easy to obtain, estimate or compute, and store.

2. Statistical learning. Given the database of existent customers, a statistical model exploring the relationship between customers' features and expenditure is performed for each group of customers assigned to a specific depot. Considering several groups, it is allowed the existence of a different trend in each one. A high number of methodologies are available to carry out regression analysis (Hastie et al., 2009; Lantz, 2013).

3. Prediction of expenditure for new customers. Once a methodology has been selected and the different functions have been fitted, the expenditure is predicted for each new customer given his/her features if assigned to each depot.

4. Assignment of customers to depots. In order to perform an efficient and feasible assignation, it is necessary not only to consider the predicted expenditure but also the distribution costs, the maximum number of vehicles per depot, and their capacity. Taking a decision for each customer individually may provide non-feasible and poor-quality solutions. Consequently, a global and iterative strategy is presented in which customers are selected one at a time to be assigned to a specific depot. It prioritizes the assignments of those customers that have associated a relatively high expected benefits only for a particular depot, and is based on the procedure developed in Juan et al. (2015c). In particular, the following steps are proposed:

   - For each depot k and customer i,
     - Compute the expected benefits μ_i^k as the difference between the predicted expenditure p_i^k and the distribution costs c_i^k (computed as the cost of moving from k to i).
     - Compute the difference between the expected benefits of assigning i to k and the maximum expected benefits of assigning i to a depot l other than k, i.e.:

$$s_i^k = \mu_i^k - max_{l \in V_d \setminus \{k\}} \mu_i^l \quad \forall i \in V_c, \ \forall k \in V_d$$

This measure is referred to as "marginal savings". Accordingly, s_i^k will be high in the case customer i reports relevant expected benefits only if assigned to k, low (in absolute terms) if the expected benefits are similar for k and at least one other depot, presenting both depots the highest expected benefits, and very low (negative) when there is at least one depot where the expected benefits are larger than those estimated for k.

- For each depot k, create a priority list of customers and sort it in descending order according to the marginal savings s_i^k.
- Create a list of unassigned customers. Then, select a depot and choose the next customer to assign from its priority list. Update the list of unassigned customers and repeat these steps while there are unassigned customers. Different policies may be applied to determine which depot selects the next customer, as: *(i)* allowing the depot with the highest remaining capacity to choose, *(ii)* using a round robin-based criterion, or *(iii)* selecting it randomly.

5. Routing. Having an assignment map, the MDVRP can be solved as a set of independent CVRPs. However, the most important challenge when addressing a MDVRP instance is the interrelation between assignation and routing. Therefore, algorithms are required to take the decisions associated to both phases 'simultaneously'. Thus, instead of finding an optimal or near-optimal solution for the customer-to-depot assignment phase and then use this unique solution as a starting point to solve the routing phase, an iteration process combines 'good' and fast computed solutions for the first stage with 'good' and fast computed solutions for the second one in order to find a near-optimal solution for the overall problem.

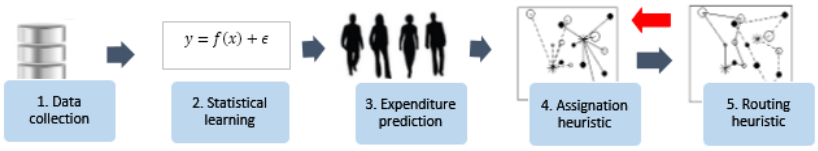


FIGURE 6.6: Scheme of the proposed approach for the MDVRP-HD.

Detailed algorithm

Figure 6.7 summarizes the proposed approach, highlighting the main differences between the classical version of the problem and the proposed one.

Since the phase of data collection is company-specific, it is assumed to be already done. The second and the third phases are related to the development and use of predictive statistical learning models. First, the database of existent customers is split into two subsets: a training set, which will be used to build the models, and a test set, to assess their performance. These subsets are generated by means of random sampling: 75% of customers are assigned to the training set and 25% to the test set. Having different alternatives to explore the relationship between expenditure and customers' features, three well-known methodologies are employed in the experiments: multiple linear regression (MLR), multi-layer feedforward network (MFN), and model tree.

- Regarding MLR, the ordinary least squares method is applied to estimate the parameters, and the stepwise regression approach with a bidirectional elimination procedure is chosen to perform the variable selection.
- The MFN with one hidden layer is considered. The number of hidden units (4, 5, 6, 7, or 8) and the decay value for regularization (0.2, 0.3, 0.4, 0.5 or 0.6) are set using 10-fold cross validation based on the metric R^2 (Kuhn, 2008). The back propagation method is employed to estimate the parameters.
- The algorithm selected to implement a model tree is the standard M5P (Wang and Witten, 1996).

The mean squared error (MSE) for each model (the number of models is the number of depots multiplied by the number of methodologies tested) using the same problem instance is computed. The total MSE is computed by aggregating the values of the models corresponding to the same methodology. In the experiments, the methodology selected is the one with the lowest total MSE. Thus, during the third phase, the expenditure that each new customer would make if he/she was assigned to each one of the depots is predicted using the selected methodology and the customer's features.

For the assignation and the routing phases, an existing methodology described in Juan et al. (2015c) has been adapted. The authors propose an efficient algorithm based on the ILS metaheuristic framework. Firstly, an initial solution is generated assigning customers to depots according to the marginal savings (only the distribution costs are considered) and designing the routes by implementing the classical CWS heuristic. Afterwards, an iterative procedure is started in which the base solution (the initial solution in the first iteration) is perturbed. If the new solution is better than the base solution, then the latter is replaced. In case no improvement is achieved, a Demon-based acceptance criterion is considered to avoid entrapment at local optimum. These steps are repeated until a termination condition is met. Finally, the top best solutions are improved by means of a post optimization process, and the best one is returned. The described algorithm includes biased randomization techniques to further diversify the search (Juan et al., 2009c). They are implemented both in the assignation phase, to randomize the sorted priority list of customers of each depot in such a way that the reasoning behind the sorting is not erased but many orderings are provided, and in the routing phase, where the CWS heuristic is randomized.

6.4.2 Computational experiments

An algorithm based on the described approach has been implemented and employed to solve a number of generated instances. The computational experiments compare the results of our approach for the analyzed version of the MDVRP and for the classical version (i.e., the one assuming homogeneous depots).

Set of instances

A total of 15 instances have been generated. Each of them consists in three datasets: the first two gather data concerning existent and new customers, respectively, and the third includes depots' locations and information related to restrictions. Regarding data of existent customers, four variables have been created: age (a discrete variable following a Uniform distribution with parameters 16 and 80), sex (a categorical variable with two equally probable values), estimated income (it follows a Normal distribution with a mean of 1500 and standard deviation of 300), and preferred article (a categorical variable including four equally probable values). Initially, each customer has been assigned to his/her closest depot, while the expenditure level has been determined by a given function that depends on the depot, the aforementioned variables and a white noise term. For a total of 100 new customers, the variables age, sex, estimated income and preferred article have been generated using the same distributions. Customers' and depots' locations have been randomly generated in a square of 100 x 100. In order to simplify the instances' generation, Euclidean distances are employed as distribution costs. Different values have been chosen for the number of depots, existent customers and vehicles, the maximum cost per route and vehicles' capacity. This information is shown in Table 6.7.

Test

Each instance has been adapted by modifying the expenditure of existent customers to analyze the following scenarios: *(1)* low ratio (LR), the average ratio between average expenditure of existent customers and average distribution costs is similar; *(2)* medium ratio (MR), average expenditure is relatively higher than average distribution costs; and *(3)* high ratio (HR), average expenditure is much higher than average distribution costs. The target ratio has been reached multiplying expenditures by a coefficient. The analysis of these scenarios will allow the comparison of the expected benefits (expected incomes, defined as the sum of predicted expenditures, minus distribution costs) associated to solutions considering only distribution costs and those taking into account also customer preferences (predicted expenditure), thus exploring the consequences of having different weights of expenditure in the objective solution. For the first scenario, it is expected that the gap

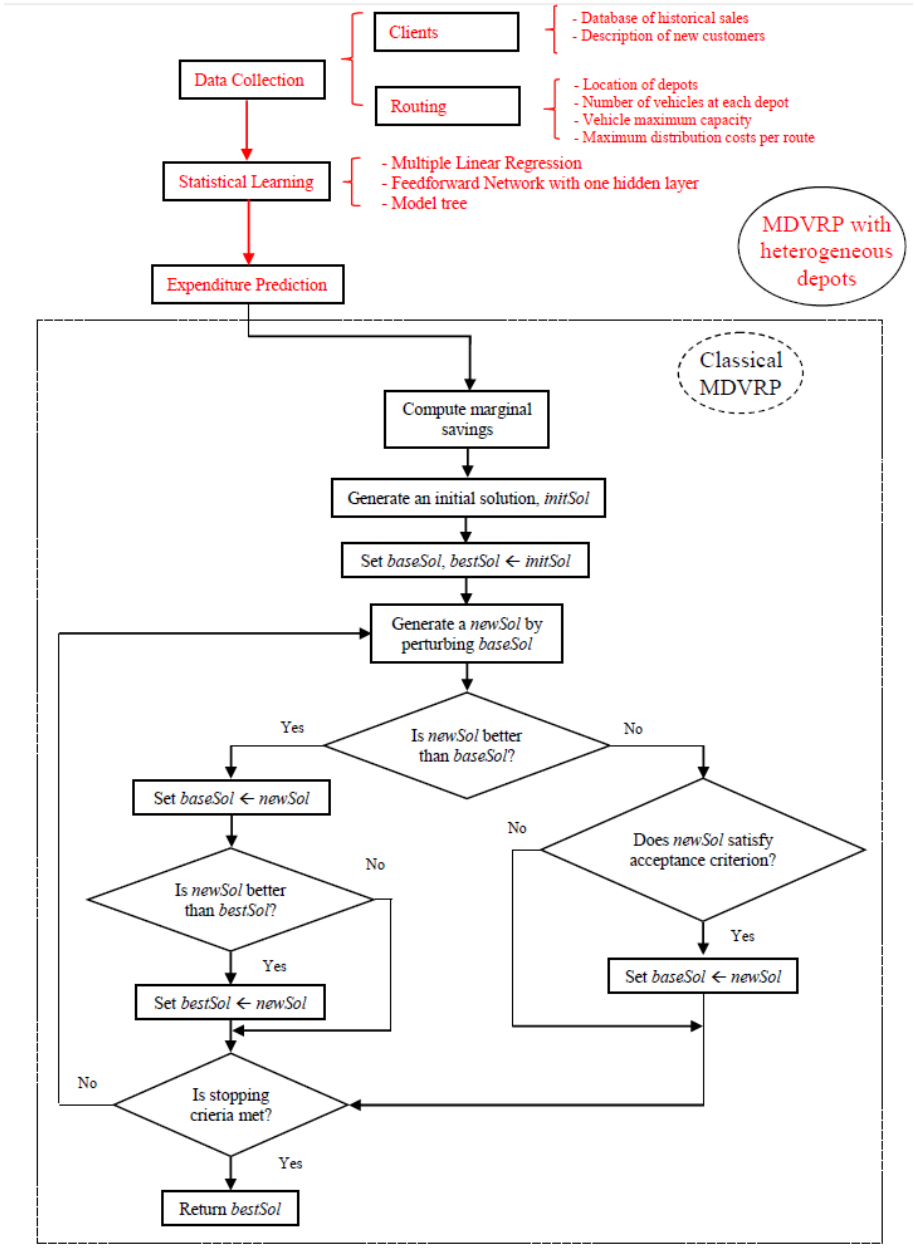


FIGURE 6.7: Flowchart of the proposed approach for the MDVRP-HD.

between distribution costs will be low (i.e., solutions are expected to be relatively similar). Likewise, it is expected that this gap will be higher as the ratio increases. Similarly, it is also expected that the higher the ratio, the higher the gap between the expected benefits of the solutions. The code has been implemented with Java and R (Team, 2008). The ILS process runs for 4,000 iterations, and all executions are solved for 10 different seeds. Only the best values obtained after the 10 runs are reported.

Results

Tables 6.8, 6.9 and 6.10 show the results. The information gathered is the following: instance name; methodology selected for prediction; distribution costs, expected incomes, expected benefits and time associated to the best solution found considering only distribution costs (classical MDVRP) and to the best solution found when maximizing expected benefit (MDVRP with heterogeneous depots); and gaps between distribution costs, expected incomes and expected benefits

Instance	Numb. depots	Numb. existent cust.	Numb. vehicles	Vehicle capacity	Max. cost
1	3	300	3	250	200
2	3	300	3	225	200
3	3	300	3	225	150
4	3	300	3	225	200
5	3	300	3	200	150
6	3	400	3	350	225
7	3	400	3	300	200
8	3	400	3	200	175
9	5	400	4	325	175
10	5	400	4	200	150
11	5	400	4	275	175
12	5	400	4	275	150
13	5	400	4	225	200
14	5	400	4	175	125
15	5	400	4	250	175

TABLE 6.7: Description of the generated instances.

of both solutions.

		Traditional (1)				Rich(2)				Gaps(2-1)		
Inst.	Meth.	Dist. cost	Exp. inc.	Exp. ben.	Time	Dist. cost	Exp. inc.	Exp. ben.	Time	Dist. cost	Exp. inc.	Exp. ben.
p01.1	MLR	898.6	961	62.4	82	930.6	1006	75.4	123	31.9	45.0	13.1
p02.1	M5P	834.3	943	108.7	112	834.5	947	112.6	335	0.1	4.0	3.9
p03.1	MFN	944.0	911	-33.0	143	964.4	939	-25.4	159	20.4	28.0	7.6
p04.1	MFN	891.8	852	-39.8	79	923.4	884	-39.4	165	31.6	32.0	0.4
p05.1	MFN	909.7	824	-85.7	189	914.4	829	-85.4	66	4.8	5.0	0.2
p06.1	MFN	868.5	1425	556.5	655	870.2	1429	558.8	613	1.7	4.0	2.3
p07.1	MFN	923.4	1073	149.6	103	925.7	1093	167.3	383	2.3	20.0	17.7
p08.1	M5P	898.2	867	-31.2	105	900.9	872	-28.9	122	2.7	5.0	2.3
p09.1	MLR	1039.2	2008	968.8	91	1127.5	2218	1090.5	33	88.3	210.0	121.7
p10.1	MFN	1029.6	1404	374.4	63	1062.5	1462	399.5	40	32.9	58.0	25.1
p11.1	MLR	880.7	1469	588.3	47	939.1	1609	669.9	464	58.4	140.0	81.6
p12.1	MFN	1858.4	1699	-159.4	108	1864.2	1709	-155.2	328	5.8	10.0	4.2
p13.1	MLR	1428.3	1495	66.7	437	1568.0	1691	123.0	144	139.6	196.0	56.4
p14.1	MFN	930.0	1163	233.0	43	930.0	1163	233.0	40	0.0	0.0	0.0
p15.1	M5P	1268.1	1401	132.9	374	1375.0	1512	137.0	59	107.0	111.0	4.0
								Average		35.2	57.9	22.7

TABLE 6.8: Table of results for the MDVRP-HD instances considering a low ratio.

6.4.3 Analysis of results

Given the flexibility of neural networks, and despite the basic topology and parameter fine-tuning, and the medium size of the training set, they have been selected to solve more than half of the instances (57.8%). MLR has provided the best results in a high number of cases (31.1%).

The gaps related to the distribution costs and the expected incomes are strictly positive except in one case. It confirms the trade-off decision-makers face between both measures; that is to say, higher distribution costs are required to obtain an increase in expected incomes. Regarding the gap of expected benefits, it is strictly positive for all instances except for two where both solutions are equal. Therefore, attempting to achieve the highest benefits studying only distribution costs in instances with heterogeneous depots results in sub-optimal solutions. As expected, all average gaps increase with the ratio, i.e., the difference between solutions (in terms of distribution costs, expected incomes or expected benefits) is positively correlated to the average expenditure for fixed average distribution costs. However, this rule does not apply for all cases. In some of them, despite the fact that the gap of expected incomes increases, so does the gap of distribution costs. As a consequence, the gap of expected benefit may be reduced.

		Traditional (1)				Rich(2)				Gaps(2-1)		
Inst.	Meth.	Dist. cost	Exp. inc.	Exp. ben.	Time	Dist. cost	Exp. inc.	Exp. ben.	Time	Dist. cost	Exp. inc.	Exp. ben.
p01.2	MLR	925.3	1383	457.7	277	978.0	1483	505.0	173	52.7	100.0	47.3
p02.2	MLR	901.2	1334	432.8	301	921.9	1385	463.1	254	20.7	51.0	30.3
p03.2	MLR	959.3	1405	445.7	134	979.1	1438	458.9	89	19.8	33.0	13.2
p04.2	MFN	942.5	1280	337.5	124	947.8	1292	344.3	101	5.3	12.0	6.7
p05.2	MFN	919.0	1264	345.0	51	921.3	1269	347.8	221	2.3	5.0	2.7
p06.2	MFN	945.6	2103	1157.4	106	948.6	2122	1173.4	327	3.1	19.0	15.9
p07.2	MFN	962.8	1581	618.2	394	992.3	1617	624.7	139	29.5	36.0	6.5
p08.2	MFN	969.9	1302	332.1	300	969.9	1302	332.1	296	0.0	0.0	0.0
p09.2	MFN	1169.6	2897	1727.4	36	1336.1	3335	1998.9	173	166.5	438.0	271.5
p10.2	MFN	1165.1	2109	943.9	161	1222.9	2222	999.1	97	57.8	113.0	55.2
p11.2	MLR	1001.8	2212	1210.2	80	1054.4	2288	1233.7	253	52.5	76.0	23.5
p12.2	MFN	1050.0	2571	1521.0	75	1070.5	2620	1549.5	41	20.6	49.0	28.4
p13.2	MLR	1633.4	2178	544.6	106	1778.2	2446	667.8	270	144.8	268.0	123.2
p14.2	MFN	1020.2	1703	682.8	63	1026.8	1717	690.2	67	6.6	14.0	7.4
p15.2	M5P	1419.6	2090	670.4	69	1560.2	2257	696.8	106	140.5	167.0	26.5
								Average		48.2	92.1	43.9

TABLE 6.9: Table of results for the MDVRP-HD instances considering a medium ratio.

		Traditional (1)				Rich(2)				Gaps(2-1)		
Inst.	Meth.	Dist. cost	Exp. inc.	Exp. ben.	Time	Dist. cost	Exp. inc.	Exp. ben.	Time	Dist. cost	Exp. inc.	Exp. ben.
p01.3	MLR	1060.3	1930	869.7	199	1153.7	2132	978.3	42	93.4	202.0	108.6
p02.3	M5P	1070.7	1803	732.3	253	1097.0	1864	767.0	174	26.3	61.0	34.7
p03.3	MFN	1042.7	1864	821.3	23	1067.1	1923	855.9	162	24.4	59.0	34.6
p04.3	MFN	1043.2	1701	657.8	54	1080.5	1755	674.5	393	37.2	54.0	16.8
p05.3	MFN	994.0	1621	627.0	174	1011.0	1657	646.0	68	17.0	36.0	19.0
p06.3	MFN	1068.1	2856	1787.9	109	1102.7	2906	1803.3	208	34.6	50.0	15.4
p07.3	MFN	1064.1	2115	1050.9	152	1081.2	2139	1057.8	71	17.1	24.0	6.9
p08.3	M5P	1069.6	1741	671.5	32	1069.6	1741	671.5	261	0.0	0.0	0.0
p09.3	MLR	1420.5	4269	2848.5	37	1690.6	4825	3134.4	138	270.1	556.0	285.9
p10.3	MFN	1434.8	2913	1478.2	113	1734.8	3396	1661.2	33	299.9	483.0	183.1
p11.3	MLR	1238.0	3020	1782.0	25	1486.3	3407	1920.7	265	248.3	387.0	138.7
p12.3	MFN	1195.7	3385	2189.3	37	1216.1	3452	2235.9	125	20.3	67.0	46.7
p13.3	MLR	1843.3	2801	957.7	79	2321.4	3387	1065.6	101	478.1	586.0	107.9
p14.3	MFN	1198.9	2297	1098.1	17	1251.0	2351	1100.0	23	52.1	54.0	1.9
p15.3	M5P	1416.0	2086	670.0	164	1595.5	2311	715.6	210	179.5	225.0	45.5
								Average		119.9	189.6	69.7

TABLE 6.10: Table of results for the MDVRP-HD instances considering a high ratio.

The results are summarized in Figures 6.8. The boxplots on the left show the expected benefits per scenario and version of the problem: considering heterogeneous depots (rich) and assuming homogeneous ones (traditional). Even if the medians associated to each ratio level do not differ significantly, the third and second quartile values do present a higher value for the extended version of the problem. This behavior is caused by the long right tails of the corresponding distributions, which indicate that for some instances the rich version results in better solutions in terms of expected benefits. The second figure displays the variables in which expected benefits are decomposed per scenario and considering the rich version. It can be observed that differences of expected benefits between scenarios are mainly due to differences between expected incomes.

6.5 Sustainable urban freight transport

This section studies a MDVRP considering the sustainability concept as optimality criteria. The three-axis of sustainability (measured as economic, environmental and social impacts) are represented by traveling distances and times, carbon emissions and risk of accidents. These measures are monetized and aggregated. Several studies have addressed the economic impacts as a variable mainly influenced by traveling distances; therefore most existing models seek to minimize them. However, doing this does not guarantee the minimum impact because many elements such as congestion, speed limits, traffic signs and vehicles crashes make longer the time of the distribution

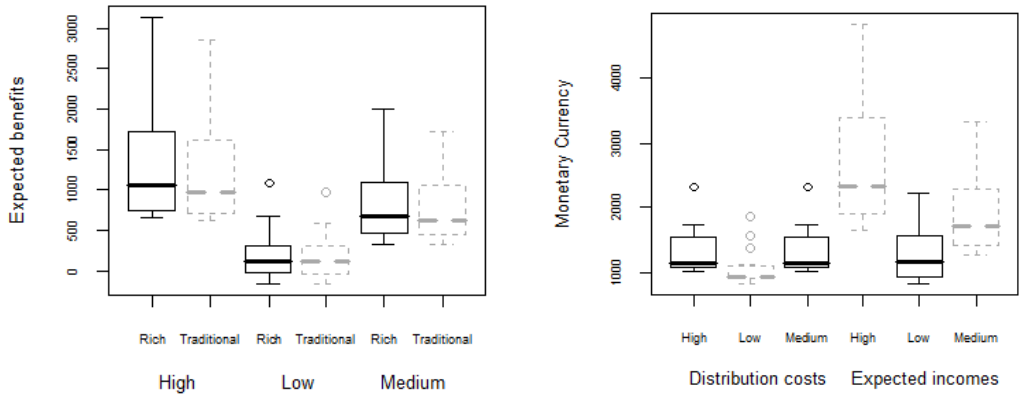


FIGURE 6.8: Boxplots of the expected benefits for the MDVRP-HD instances per scenario and version (left), and of the distribution costs and expected incomes for the rich version (right).

routes (Wang et al., 2016). In fact, the shortest paths in urban zones tend to have more traffic signs since these are the most frequented and, as a result, main streets may be the slowest paths.

- *Economic dimension:* It is composed by the classical measures total traveling times and distances, which are monetized based on the driver wage, vehicle fixed cost and oil price.
- *Environmental dimension:* CO2 emissions estimates assume that the internal combustion process of vehicles burns the carbon of the fuel and it is released as carbon dioxide. Thus, emissions are assumed to depend on fuel consumption. The fuel consumption is estimated as suggested in Kuo (2010) and Zhang et al. (2015).
- *Social dimension:* Accidents are an externality caused by speed variations on roads, among other factors. These variations represent the state and stability of the roads, and are associated to an accident risk for pedestrian and vehicles (Wang et al., 2016).

6.5.1 Methodology

The methodology proposed (Algorithm 8) is based on the VNS metaheuristic. The inputs are the problem instance to solve and the number of neighborhoods considered (K). It is usual to set K to two or three, and to design nested neighborhoods.

First, an initial solution is generated and stored in *initSol* and *baseSol*. Then, the cost of all the impacts associated are computed. *bestSol* will store the best solution found. An outer loop is started, which sets the current neighborhood to the first one. Inside, another loop builds and assesses new solutions. Within this loop, the base solution is initially shaken (i.e., it is partially destroyed and reconstructed in a random way), generating a solution from the k-th neighborhood of *baseSol*. The total cost of this solution (*newSol*) is computed (Algorithm 9). The variable *rpd* measures the relative percentage difference between the total cost of *newSol* and *baseSol*. If there is an improvement (i.e., $rpd < 0$), a local search is applied to *newSol*, the resulting solution is copied into *baseSol*, and the current neighborhood is set to the first. In addition, *bestSol* is updated if it applies. This constitutes a descendent phase aimed to find a local minimum. Otherwise, *newSol* is accepted and the current neighborhood is set to the first with a probability of $exp(-rpd)$. This acceptance criterion, first proposed in Hatami et al. (2015), aims to avoid entrapment at local optimum. It is based on the criterion of the simulated annealing metaheuristic but is simpler and has no parameters. In case of not accepting *newSol*, the next neighborhood is analyzed (i.e., k is set to $k + 1$). The inner loop is executed until the last neighborhood is explored (i.e., $k = K$). Finally, *bestSol* is returned.

The generation of solutions for the MDVRP has two sequential and interrelated stages: *a)* the assignment of customers to depots, and *b)* the design of distribution routes for each depot. Both stages employ biased randomization techniques. The first stage relies on a measure called "marginal savings" (Juan et al., 2015c), which is computed for each pair depot-customer as follows: the distance between each depot and the customer is obtained, and the difference of assigning

Algorithm 8 Approach for the MDVRP considering externalities.

```
1:  procedure MDVRP WITH SUSTAINABILITY INDICATORS (inputs, impactsParameters)
2:      initSol ← genInitSol (inputs) # generate solution based on the BR-CWS heuristic
3:      baseSol ← clone (initSol)
4:      computeTotalCost(baseSol, impactsParameters)
5:      bestSol ← clone (baseSol)
6:      while (stopping criterion is not met) do
7:          k ← 1
8:          while (k ≤ K) do
9:              newSol ← shake(baseSol, k) # destruction-construction stages
10:             computeTotalCost(newSol, impactsParameters)
11:             rpd ← (getTotalCost(newSol) - getTotalCost(baseSol))/getTotalCost(baseSol)· 100
12:             if (rpd < 0) then # newSol improves baseSol
13:                 newSol ← localSearch(newSol)
14:                 baseSol ← newSol
15:                 k ← 1
16:                 if (getTotalCost(newSol) - getTotalCost(bestSol) < 0) then
17:                     bestSol ← newSol
18:                 end if
19:             else
20:                 u ← generateU()
21:                 if (u < exp(−rpd)) then #acceptance criterion
22:                     baseSol ← newSol
23:                     k ← 1
24:                 else
25:                     k ← k + 1
26:                 end if
27:             end if
28:         end while
29:     end while
30:     bestSol ← localSearch(bestSol)
31:     return bestSol
32: end procedure
```

Algorithm 9 Function to monetize the impacts of a given solution.

```
1:  procedure COMPUTE TOTAL COST(MDVRPSol, impactsParameters)
2:      distance ← 0
3:      time ← 0
4:      emissions ← 0
5:      social ← 0
6:      for each (cvrpSol in MDVRPSol) do
7:          distance ← distance + getDistance(cvrpSol)
8:          time ← time + getTime(cvrpSol)
9:          for each (edge in cvrpSol) do
10:             emissions ← emissions + getDistance(edge)/getKPL(edge)
11:             social ← social + getDistance(edge) · getLoad(edge, cvrpSol)
12:         end for
13:     end for
14:     distanceCost ← distance· getDistUnitCost(impactsParameters)
15:     timeCost ← time· getTimeUnitCost(impactsParameters)
16:     emissionsCost ← emissions· getEmissionsUnitCost(impactsParameters)
17:     socialCost ← social· getSocialUnitCost(impactsParameters)
18:     totalCost ← distanceCost + timeCost + emissionsCost + socialCost
19:     return totalCost
20: end procedure
```

the customer to the specific depot instead of the closest depot among the others is computed. A priority list of customers is created for each depot and sorted according to the marginal savings. Thus, high marginal savings are prioritized, since assigning the corresponding customer to another depot (which would be farther) could lead to a poor-quality solution. These lists are randomized

assigning probabilities according to a Geometric distribution. Three different policies are iteratively applied to choose the depot to select the next customer to be assigned: *i)* all depots choose the first node in their list at a time, following consecutive turns (known as round robin criteria); *ii)* randomly; *iii)* the depot with the highest remaining capacity is selected. Thus, using biased randomization and different policies promotes the generation of different assignation-maps. The second stage is based on the randomized version of the CWS saving's heuristic (Juan et al., 2011a), which also depends on a Geometric distribution applied to the savings to iteratively choose one merge among all possible. However, the classical distance-based savings are replaced by "rich savings" including all costs.

The performance of each solution is computed and the sum of the costs associated to the impacts considered: economic, environmental and social. The shaking procedure randomly selects a percentage p_k of customers to be assigned to a different depot. Afterwards, the procedure to construct solutions is applied to repair the solution. This movement is introduced to diversify the search. This search is guided by the base solution, since the shaking procedure applied at each iteration works with that solution. It is set to the initial solution at the beginning and replaced by the new solution if the acceptance criterion is met. The stopping criterion is based on the number of iterations. Two local searches are used: the first is applied to solutions improving the current base solution and is based on the classical 2-opt operator defined for the CVRP (Lin, 1965), while the second is a routing extensive improving search described in Juan et al. (2011a), and applied only to the best solution found at the end. More details for specific procedures can be found in the references provided in this section.

6.5.2 Computational experiments

The algorithm proposed has been implemented in JAVA and run on a personal computer with 8 GB of RAM and an Intel Core i7 of 1.8 GHz. In order to test it, illustrate its use and the analysis of results that may be carried out, 4 MDVRP benchmark instances (p10, p11, p12 and p13) called here instance 1, 2, 3 and 4, respectively, are employed. They have been extensively used (see Vidal et al., 2012; Escobar et al., 2014).

Each instance has been adapted as follows. Vehicles' efficiency parameters are based on a type of light duty vehicle used for freight distribution in urban zones. We have used the cost coefficients of Zhang et al. (2015) for CO2 emissions (0.1 USD/L). Regarding the traveling time cost, it is defined by Koç et al. (2014) as the sum of a vehicle fixed cost and driver wage, which are set to 1.4 USD/h and 6.3 USD/h, respectively. The traveling distance cost is based on the price of fuel (1.1 USD/L) and the average miles per fuel liter (5.56 km/L). Delucchi and McCubbin (2010) propose an the interval $[1 \cdot 10^{-4},\ 1.3 \cdot 10^{-3}]$ USD per kg-km for the coefficient to estimate the social cost. Without loss of generality, times t_{ij} are generated from distances d_{ij} using this formula $t_{ij} = \alpha \cdot d_{ij} + \varepsilon_{ij}$, where α is a constant based on an estimated speed ($\alpha^{-1} = 35$ km/h) and ε_{ij} represents external factors that define the correlation between traveling time and distance. It is set to follow a truncated Normal distribution with a lower bound and mean equal to 0 and a standard deviation equal to 3.5, 2, and 0.5. These deviations are set in order to get a correlation around 0.5, 0.7 and 0.9, which may represent a high, medium and low congested zone, respectively. Thus, three scenarios are generated per instance. For example, for d_{ij}=10 km, t_{ij} would fall in the following intervals considering the different standard deviations and a probability of 95%: (0.59, 1.69), (0.64, 5.06), and (0.69, 8.42).

Each instance has been solved 10 times (employing a different seed for the random number generator) and only the best solutions are reported. 300,000 iterations are considered. The parameter fine-tuning is performed by using design of experiments and testing reasonable ranges. The parameters for the Geometric distributions related to the allocation and the routing process are randomly chosen in the intervals (0.5, 0.8) and (0.1, 0.2), respectively. The degree of shaking, which defines the neighborhoods, is set to 10%, 30% and 50% (for the first, second and third neighborhood, respectively).

The experimentation process consists on analyzing how the solution space changes according to the optimization criterion and how it influences the other indicators. Thus, five options are considered: optimization criterion is based on minimizing each component of the objective function or the sum of them. In a real-life application, the choice will depend on the particular interests of the decision-maker.

6.5.3 Analysis of results

This section analyzes the solutions found considering all the indicators or each one of them as optimization criterion. This comparison aims to determine a solution subspace representing an equilibrium between the economic, environmental, and social dimensions.

Table 6.11 shows the total cost of the best solutions found according to the objective pursued for each instance and scenario. As described before, the total cost is computed as the sum of the costs associated to traveling distance, traveling time and CO2 emissions, and the social cost. As expected, the best solution in terms of total cost is the solution that seeks to minimize the total cost. In instance 1 and 2, the solution minimizing the traveling time matches the solution minimizing the total cost, which means that these objectives converge to the same solution. Similarly, the same solution ensures the minimum traveling distance cost and emissions cost (this is due to the way in which the emissions are estimated). Obviously, the total cost is higher in the zones where the traveling time and the traveling distance have a low correlation (i.e., in congested zones, based on the description of the scenarios). The solution with the minimum social cost is the most expensive, because the other costs are significantly increased.

TABLE 6.11: Total cost by scenario, instance and optimization criterion.

		Scenarios			
		Low	Medium	High	
Instance	Objective	Total Cost	Total Cost	Total Cost	Run Time (s)
Instance 1	Total Cost	7597.4	5669.1	3763.3	1665.8
	Distance	8601.8	6054.1	3763.3	1572.9
	Time	7597.4	5770.8	3867.3	1688.3
	CO2 Emissions	8601.8	6054.1	3763.3	1572.9
	Social cost	8686.3	6393.7	3977.7	1485.0
Instance2	Total Cost	7625.8	5979.4	3645.0	1368.4
	Distance	9087.9	6392.3	3645.0	1625.2
	Time	7625.8	6060.9	3741.2	1603.2
	CO2 Emissions	9087.9	6392.3	3645.0	1625.2
	Social cost	9096.5	6651.7	3898.9	1130.3
Instance 3	Total Cost	2475.6	1913.3	1197.9	200.2
	Distance	2949.0	1944.5	1199.6	190.8
	Time	2475.6	1949.0	1197.9	198.5
	CO2 Emissions	2949.0	1944.5	1199.6	190.8
	Social cost	2979.8	1999.8	1241.6	186.0
Instance 4	Total Cost	2757.8	1904.3	1217.0	185.9
	Distance	2871.1	1913.3	1217.0	187.3
	Time	2813.3	1927.9	1222.7	189.3
	CO2 Emissions	2871.1	1913.3	1217.0	183.9
	Social cost	3144.2	2103.1	1385.6	182.8

Regarding the social cost, it is important to determine the customer sequence and the direction of the route. Figure 6.9 illustrates and quantifies their effect on the total cost of a given route. Accordingly, high-quality solutions visits first the customers with higher demands, minimizing the amount of freight transported over long stretch of roads. On the other hand, the scenario influences the total cost. Table 6.11 suggests that the gap between solutions with minimum total cost and minimum social cost is higher in congested zones. This happens because minimizing the social cost involves reducing the traveling distance, which leads to optimize also the traveling time if there is a high correlation between time and distance.

Figures 6.10 and 6.11 provide information regarding the behavior of solutions for each scenario. The first represents the average weight of each cost component per scenario considering the four instances. It can be observed that traveling time represents the main cost and its magnitude is the most sensitive to the scenario. Figure 6.11 shows the ranges of total cost and its components per scenario for instance 1. In this case, the time cost increases at a higher rate than the distance cost, which causes differences among scenarios.

Table 6.12 shows the cost of each indicator when the main objective is to minimize the total cost for all instances and scenarios. The gaps reflect the difference between the solution with minimum total cost and the best solution for each indicator. For example, the solution with the minimum total cost for instance 1 in the low scenario has a social cost 9.49% higher than the

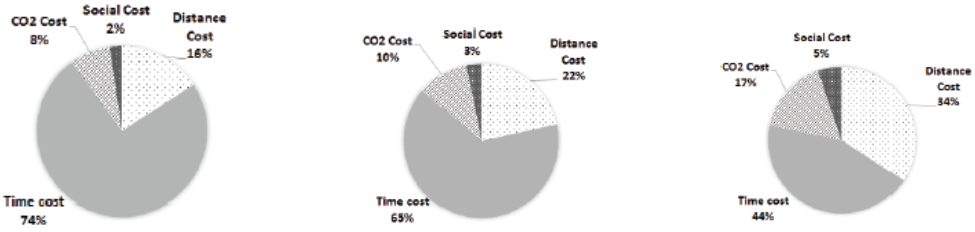
Route	1	2	3	4	5	6	7	8	9	10	11	1
Demand		70	71	56	14	50	79	57	50	22	14	
Distance		1.2	0.5	0.7	0.9	1.3	0.8	0.9	0.3	0.7	0.6	
a_{ij}		1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	
Social Cost		555.5	198.2	236.0	263.1	359.0	166.5	124.4	25.8	24.1	7.8	

Total Cost **$1,961**

Route	1	11	10	9	8	7	6	5	4	3	2	1
Demand		14	22	50	57	79	50	14	56	71	70	
Distance		0.6	0.7	0.3	0.9	0.8	1.3	0.9	0.7	0.5	1.2	
a_{ij}		1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	
Social Cost		270.5	314.2	134.1	345.4	255.0	344.5	194.1	135.9	67.7	80.5	

Total Cost **$2,142**

FIGURE 6.9: Effect of the customer sequence and the direction for a given route.



(A) Low Scenario (B) Medium Scenario (C) High Scenario

FIGURE 6.10: Weight of each sustainability component in the total cost by scenario considering all instances.

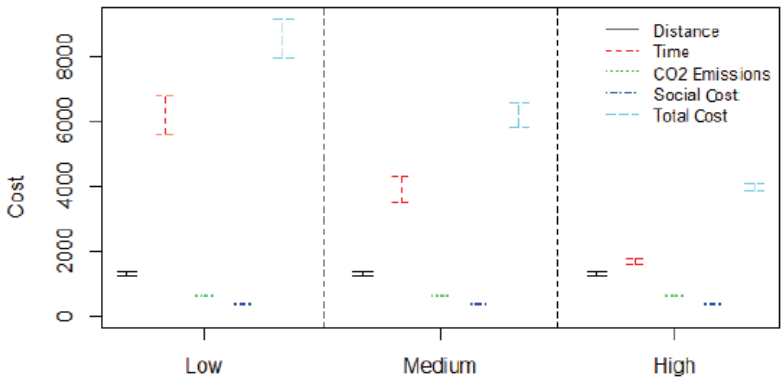


FIGURE 6.11: Total cost and component per scenario for instance '1'.

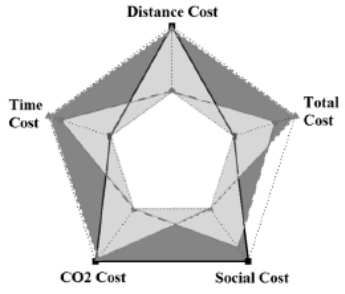
best solution found when the objective is to minimize the social cost. This table demonstrates that the solution with the minimum total cost does not tend to be the best when applying another optimization criterion.

Figure 6.12 displays radar plots for instances 1 and 4, and the scenarios low and high using the best solutions found for each indicator and the total cost. These plots identify the desirable and sustainability regions. The desirable region may be used to define an upper bound (or maximum allowable cost for each measure) and a lower bound (i.e., the white regular pentagon), which represents the ideal solution. There is no guarantee that a feasible solution exists that falls in the sustainability region. However, if one is found, it can be argued that that solution achieves a

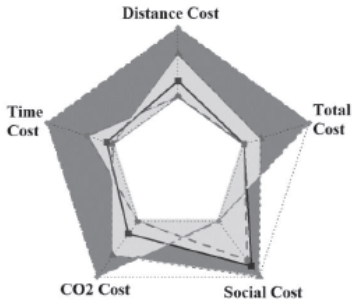
TABLE 6.12: Comparison among solutions for each instance and scenario.

Objective: Minimizing Total cost										
Scenario		Instance	Traveling distance		Traveling time		CO2 emissions		Social cost	
		(V, C, D)	Cost	Gap (%)	Cost	Gap (%)	Cost	Gap (%)	Cost	Gap (%)
Low	1	(8,249,4)	1386.35	-12.88%	5136.15	0.00%	672.10	-12.88%	402.78	-20.37%
	2	(6,249,5)	1376.06	-14.96%	5192.52	0.00%	667.11	-14.96%	390.11	-21.38%
	3	(5,80,2)	560.84	-23.09%	1624.93	0.00%	271.89	-23.09%	17.90	-31.55%
	4	(5,80,2)	460.19	-5.66%	2059.82	-1.16%	223.10	-5.66%	14.69	-28.31%
		Average	945.86	-14.15%	3503.36	-0.29%	458.55	-14.15%	206.37	-25.40%
Medium	1	(8,249,4)	1328.31	-9.07%	3325.69	-2.38%	643.96	-9.07%	371.14	-13.58%
	2	(6,249,5)	1312.04	-10.81%	3671.20	-2.96%	636.08	-10.81%	360.05	-14.82%
	3	(5,80,2)	434.16	-0.65%	1254.84	-12.01%	210.48	-0.65%	13.78	-11.05%
	4	(5,80,2)	442.88	-1.97%	1232.64	-0.26%	214.71	-1.97%	14.09	-25.23%
		Average	879.35	-5.63%	2371.09	-4.40%	426.31	-5.63%	189.77	-16.17%
High	1	(8,249,4)	1204.67	0.00%	1626.95	-4.72%	584.02	0.00%	347.65	-7.74%
	2	(6,249,5)	1177.17	-0.59%	1563.70	-3.79%	570.69	-0.59%	333.45	-8.03%
	3	(5,80,2)	435.35	-0.92%	538.16	0.00%	211.06	-0.92%	13.33	-8.06%
	4	(5,80,2)	434.16	0.00%	558.53	-0.48%	210.48	0.00%	13.78	-23.54%
		Average	812.84	-0.38%	1071.84	-2.25%	394.06	-0.38%	177.05	-11.84%

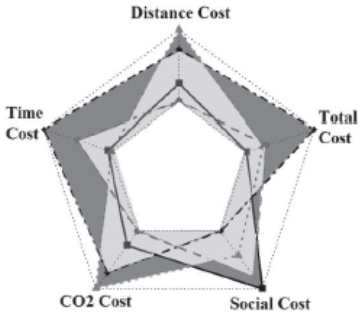
suitable balance between at least two measures. In our case, the sustainability region is a narrower area for the scenario with less nodes and a high correlation between traveling time and distance.



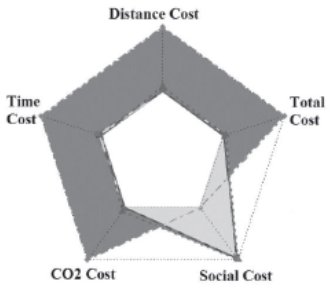
(A) Instance 1: Low Scenario



(B) Instance 1: High Scenario



(C) Instance 4: Low Scenario



(D) Instance 4: High Scenario

FIGURE 6.12: Solution spaces for decision-making considering sustainability indicators.

6.6 The WCP

The WCP (Figure 6.13) can be described on a graph $G = (V, A)$, where the set of nodes $V = V^d \cup V^f \cup V^c \cup V^b$ includes: *(i)* a set of starting and ending depots $V^d = \{0, 0'\}$ (in practice both depots could be the same), with the starting depot being the initial location of a fleet of homogeneous vehicles $K = \{1, 2, \ldots, k\}$, each of them having a capacity C; *(ii)* a set $V^f = \{1, 2, \ldots, m\}$ describing m landfills at which collected waste must be disposed at least once before visiting the ending depot; *(iii)* a set of waste containers (customers) $V^c = \{m+1, \ldots, m+n\}$ with associated waste levels $q_i > 0$ ($\forall i \in V^c$); and *(iv)* a set $V^b = \{0^*\}$ representing a virtual lunch-break node that has to be included

in each route. Each node $i \in V \setminus V^d$ has an associated time window represented by $[a_i, b_i]$ (with $0 \leq a_i < b_i$). Necessary service times for emptying any container and the duration of the lunch break are formulated as $r_i > 0$ ($\forall i \in V^c \cup V^b$). Likewise, the set $A = \{(i, j)/i, j \in V, i \neq j\}$ describes the arcs connecting any pair of different nodes. Each pair is characterized by its respective travel costs, $c_{ij} = c_{ji} \geq 0$, and travel times, $t_{ij} = t_{ji} \geq 0$. The travel time associated with going from any node $i \in V \cup V^b$ to the virtual lunch-break node (and vice versa) is equal to zero, i.e.: $t_{i0^*} = t_{0^*i} = 0$. Notice, however, that the travel cost associated with 'crossing' the lunch-break virtual node is given by the travel cost of the origin and destination nodes, i.e.: $c_{i0^*} + c_{0^*j} = c_{ij}$. The decision variables x_{ijl} ($\forall (i, j) \in A, \forall l \in K$) equal 1 if arc (i, j) is employed by vehicle l and 0 otherwise. The aim is to minimize total travel costs.

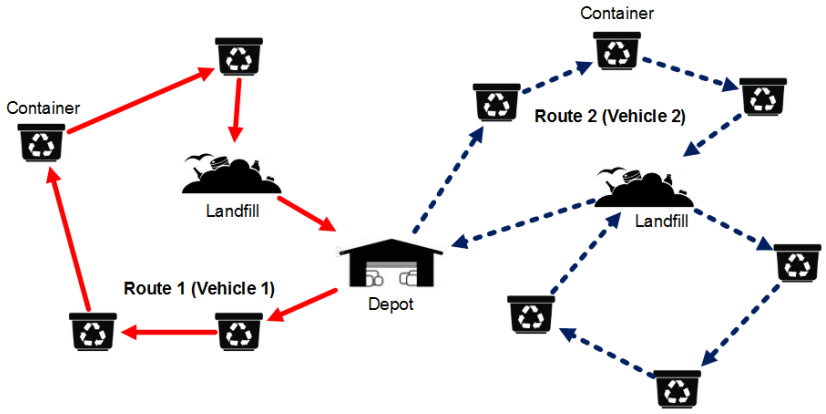


FIGURE 6.13: Representation of a WCP instance.

The following restrictions are considered: *(i)* the number of vehicles used is not predetermined, only the maximum number of available vehicles is given; *(ii)* the lunch break is automatically included in a route whenever a certain time window is reached; *(iii)* there is a maximum number of stops at containers and landfills per route; *(iv)* there is a maximum amount of waste that can be collected on a single vehicle route; and *(v)* the depot also has a time window. Methodologies for both the deterministic and the stochastic versions are presented.

6.6.1 Methodology

The WCP

A VNS metaheuristic is proposed to solve the deterministic WCP. An initial solution is obtained by applying the CWS heuristic and its biased-randomized extension. This procedure is adapted to the special case of waste collection by changing the calculation of savings values used for merging two customers i and j, originally calculated as $s_{ij} = c_{i0} + s_{0j} - c_{ij}$. In the WCP, the costs of traveling between a customer and the depot are asymmetric due to the additional landfill visit. To address this new situation, the average savings associated to each arc are employed.

Based on the initial solution $baseSol$, different neighborhood structures $N_k(\{k = 1, ..., k_{max}\})$ are created. The shaking procedures applied to create new solution structures are outlined in Table 6.13. Within each neighborhood $N_k(baseSol)$, different local descent heuristics described in Table 6.14 are randomly applied to find the local minimum of $N_k(baseSol)$. To conclude the local search phase, a quick solution improvement procedure based on a cache memory technique (Juan et al., 2013a) is implemented: the best-known order of traveling between a set of nodes establishing a sub-route –i.e., starting at the depot or a landfill and ending at a disposal site– is stored in a hash-table data structure, thus allowing new solutions to benefit from previously constructed ones. Whenever the local search phase leads to a more competitive objective function value than that of $baseSol$, $baseSol$ is updated and k is returned to its initial value of 1. If $baseSol$ cannot be improved through the local minimum of N_k, k is incremented by 1 and the next shaking operator is applied. Once each neighborhood has been constructed ($k = k_{max}$), the process is repeated until a certain predefined stopping criterion (e.g.: time, iterations, etc.) has been reached. Note that we shuffle the list of neighborhood operators every time $k > k_{max}$. A description of the VNS procedure for the deterministic WCP can be seen in Algorithm 1.

TABLE 6.13: Shaking operators for the WCP.

Operator (k)	Description
Customer Swap Inter-Route	Swaps two random customers between different routes.
2-Opt Inter-Route	Interchanges two chains of randomly selected customers between different routes.
Reinsertion Inter-Route	Inserts a random customer in a different route.
Cross-Exchange	Interchanges positions of 2-4 random, non-consecutive customers from different routes.

TABLE 6.14: Local search operators for the WCP.

Operator (LS-Scheme)	Description
Best Position Insertion	Reinserts the container with the highest objective function increase into the best available position of any route.
Re-allocate all	Iteratively calculates the objective function increase of each container and reinserts it at the best possible position.
Random Swaps	Randomly selects and interchanges two nodes (from the same or different routes) if the objective function improves.

The stochastic WCP

Waste levels cannot be predicted with full certainty when solving a more realistic stochastic version of the problem. The fact that actual waste levels in containers are only known when reaching designated pick-up points can lead to route failures whenever collected garbage exceeds the planned collection amount. In these cases, the collection vehicle needs to add an additional and expensive landfill visit to its route. The proposed simheuristic methodology (Algorithm 2) allows an estimation of the solution quality of previously created outputs using the VNS metaheuristic proposed before by integrating MCS into the solution procedure.

The methodology starts by transforming the stochastic input variables into their deterministic counterpart, which is used to establish initial WCP solutions. Even though waste levels face different levels of stochasticity, their behavior can typically be modeled according to some kind of theoretical or empirical distribution (e.g., based on historical data). This allows the (stochastic) waste levels w_i at each container i to be replaced with expected values $E[w_i]$. Using these deterministic values, an initial solution $baseSol$ is constructed. In the following, the solution quality in a stochastic environment is tested by randomly simulating the waste levels of each container i for a certain number of iterations (or simulation runs) within the predefined probability distribution. During each run the occurring route failure costs are estimated by penalizing situations in which vehicle capacities are reached before a scheduled landfill trip. More specifically, route failure costs are calculated as corrective actions to the predefined routes. Finally, the sum of all route failure costs of all simulation runs are divided by the number of simulation runs. Thus, the expected total costs of $baseSol$ consist not only of the deterministic routing costs, but rather in the addition of the deterministic routing costs with the expected route failure costs. At this stage we propose the application of a small number of iterations $shortSimIter$. On the one hand, a larger number of simulation runs lead to more reliable estimates of the stochastic route costs. On the other hand, at this stage a shorter simulation procedure can be used to keep the computational effort through the simulation reasonable.

Once $detCosts(baseSol)$, $stochCosts(baseSol)$, and $totalCosts(baseSol)$ have been defined, new deterministic solution neighborhoods are constructed and locally improved as described previously. A newly constructed solution $newSol$ is considered as promising whenever it yields lower deterministic costs than the current base solution. The behavior of each promising solution under waste level uncertainty is then evaluated by applying a short simulation run, leading to a first estimation of the total solution costs. Whenever $totalCosts(newSol) < totalCosts(baseSol)$, the current base solution is updated and k is returned to its initial value. Furthermore, the solution is stored as elite stochastic solution. With each elite solution, a more extensive simulation run is started for $longSimIter$ iterations once the metaheuristic stopping criteria has been reached. The number of stored $eliteSols$ is limited to 10.

In addition to calculating the stochastic objective function value of promising deterministic solutions, our methodology allows the estimation of a solution reliability by considering the proportion of runs where the solution plan can be implemented without any route failure. Thus,

Algorithm 1 VNS procedure for the WCP

```
1:  baseSol ← solve biased randomized CWS for the WCP
2:  while stopping criteria not reached do
3:      shuffle(ListOfShakingOperators)
4:      k ← 1
5:      repeat
6:      newSol ← shake(baseSol, k)
7:      improving ← true
8:      while improving do
9:          newSol* ← localDescent(newSol, randomLSoperator)
10:         if costs(newSol*) ≤ costs(newSol) then
11:             newSol ← newSol*
12:         else
13:             improving ← false
14:         end if
15:         cacheSubRoutes(newSol)
16:         if costs(newSol) < costs(baseSol) then
17:             baseSol ← newSol
18:             k ← 1
19:         else
20:             k ← k + 1
21:         end if
22:     end while
23:     until k > k_max
24: end while
25: bestSol ← baseSol
26: return bestSol
```

the reliability $reliab_r$ of each route r of any solution S is computed as the quotient of the number of runs in which a route failure occurs divided by the total number of simulation runs, i.e. $reliab_r = simRunsWithRouteFailue/simRuns$. Notice that each route in a solution can be seen as an independent component of a series system (i.e., the proposed solution will fail if, and only if, a failure occurs in any of its routes). Therefore, the overall reliability of a solution with R routes can be computed as $\prod_{r=1}^{R} reliab_r$.

6.6.2 Computational experiments

To test the deterministic approach, the 10 WCP benchmark instances provided by Kim et al. (2006), which were later adopted by Benjamin and Beasley (2010) and Buhrkal et al. (2012), are employed. Furthermore, the clustered instances presented by Buhrkal et al. (2012) are used. A clustering procedure is applied to nodes with the same location and time windows to change the total number of nodes. The algorithm was implemented as Java application and run on a personal computer with an Intel®Xeon™CPU E5-2630 v2 @ 2.60GHz processor. The initial solutions constructed with the biased randomized version of the savings heuristic are based on a distribution parameter randomly chosen within the range (0.4, 0.5) at each solution construction step.

The results are summarized in Table 6.15. Column (1) reports the BKS for each instance, column (2) the computational times (CT) in seconds, and column (3) the average results with 10 different random number seeds. The VNS metaheuristic is tested with two different stopping criteria. On the one hand, our best solution (achieved with 10 seeds) is reported in column (4). Furthermore, our average solution (5) and our best solution (6) with a stopping criterion of 300 seconds per instance are reported, as suggested by Benjamin and Beasley (2010). It can be seen that the proposed algorithm outperforms current BKSs by an average of -0.85% and -2.65%. Moreover, it reaches 9 new BKSs (11 with the extended algorithm running time).

Since there is a lack of stochastic benchmark instances, the non-clustered instances of Kim et al. (2006) are used as reference. The deterministic instances are then transformed into stochastic ones by using random waste levels following a log-normal distribution with expected values equal

Algorithm 2 Simheuristic approach for the WCP-SW

```
replace stochastic waste levels by expected values
baseSol ← solve biased randomized CWS for the WCP
shortSimulation(baseSol)
while stopping criteria not reached do
    k ← 1
    repeat
        newSol ← shake(baseSol, k)                          ▷ see Algorithm 1
        localSearch(newSol)                                 ▷ see Algorithm 1
        if detCosts(newSol) < detCosts(baseSol) then        ▷ Solution is promising
            shortSimulation(newSol)
            if totalCosts(newSol) < totalCosts(baseSol) then
                update(eliteSols)
                baseSol ← newSol
                k ← 1
            else
                k ← k + 1
            end if
        end if
    until k > k_max
end while
for each eliteSol do
    longSimulation(eliteSol)
    estimateReliability(eliteSol)
end for
```

to the original deterministic value. Note that the approach could be used with any other probability distribution (e.g., Weibull, gamma, etc.).

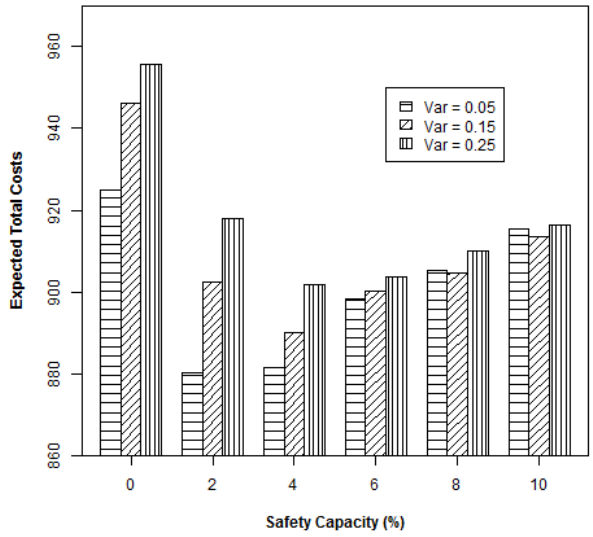
The approach is tested using low ($Var[w_i] = 0.05$), medium ($Var[w_i] = 0.15$), and high variance levels ($Var[w_i] = 0.25$) concerning the waste level distribution at any container. The number of short simulation runs is set to 500, while a more extensive simulation with 5000 runs is applied only to the elite solutions. Moreover, we propose the inclusion of vehicle safety stocks k to better deal with unexpected demands (Juan et al., 2011b). Instead of considering the complete available vehicle capacity C in the construction of the deterministic solution, a decreased capacity $C^* = C(1 - k)$ is applied. On the one hand, high levels of k will, on average, lead to higher deterministic costs, as the considered vehicle capacity during the route construction is reduced. On the other hand, it can be expected that the stochastic route failure costs will decrease. 6 different safety stock levels k are considered: 0, 0.02, 0.04, 0.06, 0.08, and 0.1.

Tables 6.16-6.18 show the deterministic costs (1), the total costs including the expected route failure penalties (2), and the related reliability calculated (3) of each tested scenario, where listed results refer to the best obtained solution according to the overall costs. The average calculation time of all scenarios was 351.92 seconds.

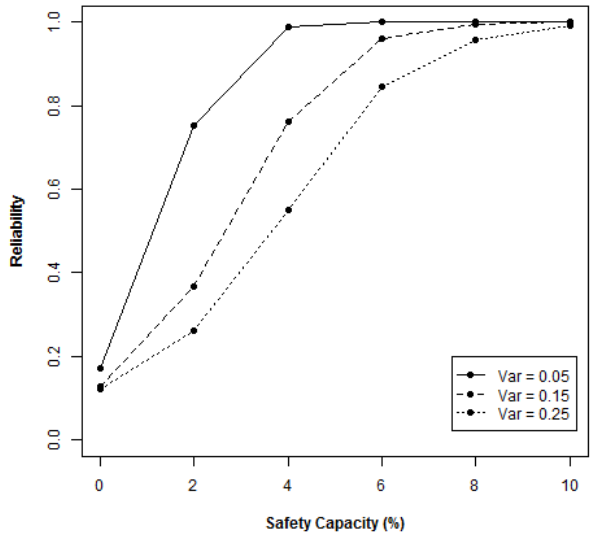
6.6.3 Analysis of results

Figure 6.14 shows the expected total costs and reliabilities for the average of all tested instances for each waste variance level/safety capacity factor combination. As can be observed, the highest total costs for each waste variance level is obtained when no safety capacity factor is considered as a result of high expected route failure costs. Furthermore, it can be seen that the lowest total costs over all instances for a low variance level are obtained with a safety capacity factor of 2%. For medium and high waste variance, a safety capacity factor of 4% seems to yield the most promising results concerning total costs. As expected however, the reliability levels increase for all variance levels as the vehicle safety capacity is increased. It can also be concluded that the inclusion of only a small safety capacity already significantly increases reliability levels (up to around 60% in the most extreme case).

A more detailed risk analysis is done in Figure 6.15, which shows a boxplot of the long simulation outputs for the three most competitive elite solutions of the Kim277 instance. In this specific



(A)



(B)

FIGURE 6.14: Expected total costs *(a)* and reliabilities *(b)* for the WCP-SW instances.

Table 6.15: Table of results for the WCP benchmark instances.

Instance	(1) BKS	(2) CT BKS (s)	(3) BKS average	(4) Our best sol[1]	(5) Our sol average[2]	(6) Our best sol[2]	(7) CT our best sol (s)	%-Gap (1)-(4)	%-Gap (1)-(6)
Kim102	174.5	3	176.03	158.61	158.64	154.62	5	-9.11	-11.39
Kim277	447.6	8	455.7	472.73	457.14	450.6	299	5.61	0.67
Kim335	182.1	10	196.49	189.79	187.36	184.22	298	4.22	1.16
Kim444	78.3	18	78.99	80.22	80.09	79.49	292	2.45	1.52
Kim804	604.1	72	650.65	603.17	601.14	593.2	300	-0.15	-1.80
Kim1051	2250.6	194	2387.7	2128.37	2119.50	2077.37	294	-5.43	-7.70
Kim1351	871.9	105	891.17	929.5	929.40	910.6	238	6.61	4.44
Kim1599	1337.5	252	1385.3	1184.67	1208.54	1182.58	292	-11.43	-11.58
Kim1932	1162.5	285	1192.2	1149.45	1169.95	1136.34	273	-1.12	-2.25
Kim2100	1749	356	1916.8	1595.48	1622.29	1603.93	293	-8.78	-8.29
Clustered Instances									
Kim86	174.5	3	176.6	155.68	158.35	155.68	10	-10.79	-10.79
Kim267	450.7	8	456.4	460.4	455.96	449.41	294	2.15	-0.29
Kim322	182.4	10	190.7	189.78	185.93	184.26	298	4.05	1.02
Kim444	78.6	18	79.2	80.22	80.09	79.49	292	2.06	1.13
Kim602	586.2	72	647.8	610.52	593.25	586.11	297	4.15	-0.02
Kim1011	2295.2	116	2370.5	2151.51	2131.00	2102.23	299	-6.26	-8.41
Kim536	850	105	850.9	885.83	877.69	850.46	292	4.22	0.05
Kim870	1170.2	252	1230.6	1156.15	1180.07	1145.83	286	-1.20	-2.08
Kim1860	1128.7	285	1180.9	1129.89	1154.48	1138.6	295	0.11	0.88
Kim1877	1594.2	266	1650.8	1620.89	1642.20	1604.33	186	1.67	0.64
Average	*868.44*	*122*	*908.27*	*846.64*	*849.65*	**833.47**	*257*	*-0.85*	*-2.65*

[1] Computational times per instance equal to column (2)
[2] Computational times per instance equal to column (7)

case the first solution seems to be the most promising one, as it has the lowest mean and the lowest quartiles. However, this is not necessarily always the case. In Table 6.19, the mean and standard deviation of the results from the long simulation concerning total costs of the three best solutions of each instance are listed. It can be concluded that the solution with the lowest mean does not always have the lowest standard deviation. This information can be used by decision-makers to select the solution that he/she prefers according to his/her risk preference. In a similar manner, our solution approach allows the consideration of different risk-aversion levels of decision takers by comparing solutions with different safety capacity levels. A more risk-averse route planner will choose to construct routes with higher safety capacity levels, which typically lead to higher routing costs while experiencing lower route failure, and vice versa.

6.7 The HSAVRP-SD

The HSAVRP-SD is defined over a complete graph $G = (N, A)$, where $N = \{0, 1, \ldots, n\}$ is a set of nodes representing the depot (node 0) and the n customers (nodes 1 to n). Each node $i \in N$ has associated a demand D_i, which is a random variable following a given probability distribution. The actual demand of a specific customer is only known when a vehicle visits her/him. The set $A = \{(i, j) : i, j \in N, i \neq j\}$ contains the arcs connecting each pair of nodes. Moreover, there is a set $F = \{1, ..., m\}$ referring to the types of vehicle. For each type $o \in F$, there are p_o available vehicles, the parameter Q_o represents the maximum load that a vehicle can carry, and U_o ($U_o \subseteq N \setminus 0$) denotes the set of customers that can be served. Each arc has associated a cost c^o_{ij} that depends on the type of vehicle. The cost of a route is the sum of the costs of its arcs and a fixed cost for using a vehicle (f_o). The goal is to design routes that satisfy all demands and minimize the total costs.

6.7.1 Methodology

The methodology proposed is a simheuristic procedure combining the ILS metaheuristic and MCS techniques. For building solutions, the successive approximations method (SAM) (Juan et al., 2014c) (Algorithm 3) is used. The description of the methodology is explained below and summarized in Figure 6.16.

TABLE 6.16: Table of results for the WCP-SW benchmark instances considering a low variance level.

Safety Capacity	**0%**			**2%**			**4%**			**6%**			**8%**			**10%**		
Instance	**(1) Det Costs**	**(2) Total Costs**	**(3) Reliab.**	**(1) Det Costs**	**(2) Total Costs**	**(3) Reliab.**	**(1) Det Costs**	**(2) Total Costs**	**(3) Reliab.**	**(1) Det Costs**	**(2) Total Costs**	**(3) Reliab.**	**(1) Det Costs**	**(2) Total Costs**	**(3) Reliab.**	**(1) Det Costs**	**(2) Total Costs**	**(3) Reliab.**
Kim102	158.78	158.78	1	158.55	158.55	1	156.14	156.15	0.9996	157.67	157.67	1	158.56	158.56	1	158.18	158.18	1
Kim277	471.48	512.35	0.2052	462.46	478.32	0.6274	486.71	487.16	0.9864	491.45	491.45	0.9998	493.89	493.89	1	494.7	494.7	1
Kim335	189.19	189.97	0.3142	187.83	188.06	0.8563	187.09	187.11	0.9944	187.65	187.65	1	189.72	189.72	1	189.66	189.66	1
Kim444	80.48	83.23	0.15	84.68	85.1	0.8476	84.52	84.55	0.9886	86.37	86.37	1	88.43	88.43	1	91.06	91.06	1
Kim804	606.48	645.04	0.0323	621.8	627.4	0.7031	624.94	625.03	0.9952	631.61	631.61	1	642.98	642.98	1	638.02	638.02	1
Kim1051	2200.05	2402.75	0	2216.32	2251.19	0.1883	2229.97	2231.21	0.9497	2313.75	2313.75	0.9996	2319.08	2319.08	1	2333.96	2333.96	1
Kim1351	924.17	1014.92	0.0158	954.2	961.61	0.756	978.95	979.31	0.9871	990.21	990.21	1	996.45	996.45	1	1021.53	1021.53	1
Kim1599	1205.23	1296.8	0.002	1209.95	1217.99	0.6641	1242.46	1242.54	0.9934	1270.23	1270.23	1	1276.18	1276.18	1	1283.12	1283.12	1
Kim1932	1144.16	1237.23	0.0008	1149.68	1152.5	0.7698	1182.4	1182.4	1	1197.76	1197.76	1	1209.53	1209.53	1	1243.92	1243.92	1
Kim2100	1599.38	1707.03	0.0002	1679.76	1683.14	0.8555	1640.45	1640.46	0.9996	1655.24	1655.24	1	1678.34	1678.34	1	1699.23	1699.23	1

TABLE 6.17: Table of results for the WCP-SW benchmark instances considering a medium variance level.

Safety Capacity	**0%**			**2%**			**4%**			**6%**			**8%**			**10%**		
Instance	**(1) Det Costs**	**(2) Total Costs**	**(3) Reliab.**	**(1) Det Costs**	**(2) Total Costs**	**(3) Reliab.**	**(1) Det Costs**	**(2) Total Costs**	**(3) Reliab.**	**(1) Det Costs**	**(2) Total Costs**	**(3) Reliab.**	**(1) Det Costs**	**(2) Total Costs**	**(3) Reliab.**	**(1) Det Costs**	**(2) Total Costs**	**(3) Reliab.**
Kim102	153.27	153.56	0.7681	154.7	154.9	0.9665	158.27	158.37	0.9968	157.67	157.73	0.9982	158.56	158.56	1	158.18	158.18	1
Kim277	462.58	524.87	0.1087	490.06	500.8	0.7097	492.65	498.32	0.8493	496.76	497.48	0.9809	494.29	494.58	0.9922	495.68	495.69	0.9998
Kim335	189.5	190.84	0.3456	187.2	187.71	0.5508	186.21	186.7	0.8249	187.17	187.19	0.9892	189.72	189.72	0.999	189.59	189.59	1
Kim444	80.43	85.53	0.034	84.69	86.8	0.3972	85.16	85.86	0.6948	86.37	86.47	0.9517	87.58	87.59	0.9966	90.84	90.84	0.999
Kim804	607.75	663.12	0.0117	623.64	643.96	0.3229	629.78	633.32	0.8274	630.59	630.79	0.9833	637.11	637.12	0.999	630.04	630.04	1
Kim1051	2201.89	2481.65	0	2215.63	2335.24	0.0026	2251.3	2278.71	0.2778	2309.16	2314.82	0.7669	2319.08	2319.78	0.9712	2333.96	2333.99	0.9988
Kim1351	925.63	1060.93	0.0019	950.02	988.05	0.2058	972.84	982.14	0.6875	1002.1	1002.87	0.9715	996.46	996.51	0.9978	1021.53	1021.53	0.9998
Kim1599	1202.08	1317.4	0.0006	1209.99	1248.56	0.123	1245.18	1251.18	0.6799	1270.23	1271.02	0.9613	1276.18	1276.2	0.9974	1283.12	1283.12	1
Kim1932	1144.16	1248.28	0.0003	1150.33	1171.73	0.1891	1182.4	1184.55	0.8934	1197.76	1197.83	0.9966	1209.53	1209.53	0.9996	1243.92	1243.92	1
Kim2100	1604.81	1735.66	0	1682.09	1707.87	0.1904	1640.79	1642.26	0.901	1655.24	1655.29	0.9962	1678.34	1678.34	1	1687.77	1687.77	1

TABLE 6.18: Table of results for the WCP-SW benchmark instances considering a high variance level.

Safety Capacity	**0%**			**2%**			**4%**			**6%**			**8%**			**10%**		
Instance	**(1) Det Costs**	**(2) Total Costs**	**(3) Reliab.**	**(1) Det Costs**	**(2) Total Costs**	**(3) Reliab.**	**(1) Det Costs**	**(2) Total Costs**	**(3) Reliab.**	**(1) Det Costs**	**(2) Total Costs**	**(3) Reliab.**	**(1) Det Costs**	**(2) Total Costs**	**(3) Reliab.**	**(1) Det Costs**	**(2) Total Costs**	**(3) Reliab.**
Kim102	158.56	158.57	0.9996	158.27	158.74	0.9842	158.27	158.65	0.9874	156.58	157.02	0.9821	157.21	157.26	0.9984	158.18	158.18	1
Kim277	461.1	537.7	0.0678	487.9	515.37	0.4067	500.22	509.25	0.75	499.31	500.78	0.9588	500.28	502.17	0.9507	498.1	498.7	0.9833
Kim335	189.61	192.76	0.142	186.4	187.86	0.3888	190.09	191.22	0.6926	189.13	189.35	0.8959	189.05	189.08	0.9821	192.09	192.09	0.9982
Kim444	80.43	86.86	0.0154	85.52	87.57	0.3537	85.14	86.83	0.4329	85.92	86.2	0.857	87.88	87.97	0.9524	90.27	90.31	0.9782
Kim804	606.33	670.43	0.0069	624.63	646.11	0.2675	630.17	639.85	0.5774	630.63	632.15	0.8855	643.52	643.93	0.9788	642.68	642.75	0.9968
Kim1051	2204.77	2518.54	0	2215.87	2387.04	0.0001	2247.29	2325.05	0.0238	2311.87	2334.32	0.3647	2319.08	2324.99	0.7783	2333.96	2334.86	0.9627
Kim1351	919.79	1060.65	0.001	950.02	1011.85	0.0728	984.52	1003.82	0.4638	1002.2	1006.84	0.8436	1013.53	1014.49	0.964	1021.53	1021.59	0.9978
Kim1599	1202.08	1329.94	0.0003	1209.99	1268.65	0.0384	1246.46	1263.17	0.3251	1270.81	1275.48	0.7911	1290.52	1291.06	0.9725	1294	1294.1	0.9926
Kim1932	1144.16	1255.11	0.0002	1150.33	1187.7	0.0572	1182.4	1191.29	0.6235	1197.76	1198.99	0.9375	1209.53	1209.59	0.996	1243.55	1243.55	0.9998
Kim2100	1604.81	1746.22	0	1682.09	1728.19	0.0461	1640.79	1647.58	0.6153	1655.24	1656.45	0.9279	1678.34	1678.4	0.9976	1687.77	1687.77	1

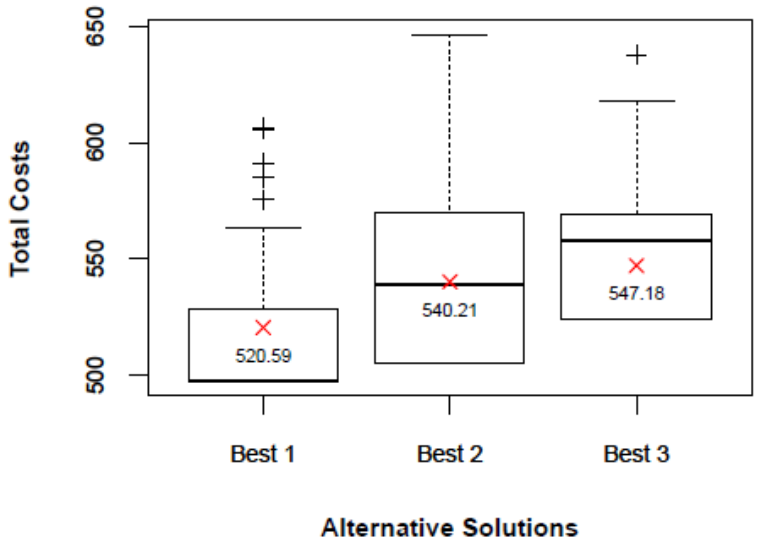


FIGURE 6.15: Boxplots of the total costs of the WCP instance 'Kim277' considering a high waste variance level and a 2% safety capacity level.

TABLE 6.19: Comparison of elite solutions for the WCP-SW.

Elite Solutions	**Best 1**		**Best 2**		**Best 3**	
Instance Name	Mean	St. Dev.	Mean	St. Dev.	Mean	St. Dev.
Kim102	157.05	3.54	157.14	3.38	157.22	3.65
Kim277	498.66	4.53	499.07	4.45	499.12	4.59
Kim335	187.84	1.81	187.96	1.84	188.25	1.85
Kim444	87.79	0.84	87.80	0.79	91.35	0.82
Kim804	633.97	5.93	634.34	5.74	635.00	5.90
Kim1051	2342.85	16.67	2343.58	15.48	2345.62	16.29
Kim1351	1009.88	26.48	1012.78	26.57	1025.50	26.54
Kim1599	1290.02	24.34	1291.67	23.40	1292.07	23.83
Kim1932	1199.85	29.77	1202.21	30.50	1245.03	30.14
Kim2100	1742.47	13.97	1742.81	14.62	1748.34	13.83

Algorithm 3 The SAM procedure

```
1:  procedure BUILDSOLUTION(customers, vehicles)
2:      globalSol ← empty
3:      nonServedCust ← customers
4:      while nonServedCust ≠ empty do
5:          vehType ← selectType(vehicles)
6:          compatCust ← getCompatibleCust(nonServedCust, vehType)
7:          sol ← solveHoSAVRP(compatCust, vehType)
8:          routes ← getRoutes(sol)
9:          numVehOfTypeK ← numberOfVehicle(vehType)
10:         if numberOfRoutes > numVehOfTypeK
11:             routes ← SelectRoutes(numVehOfTypeK, Random)
12:         end if
13:         globalSol ← addRouteToSol(routes, globalSol)
14:         vehicles ← deleteUsedVehicles(vehicles)
15:         nonServedCust ← extractCustomers(nonServedCust, globalSol)
16:     end while
17:     return globalSol
18: end procedure
```

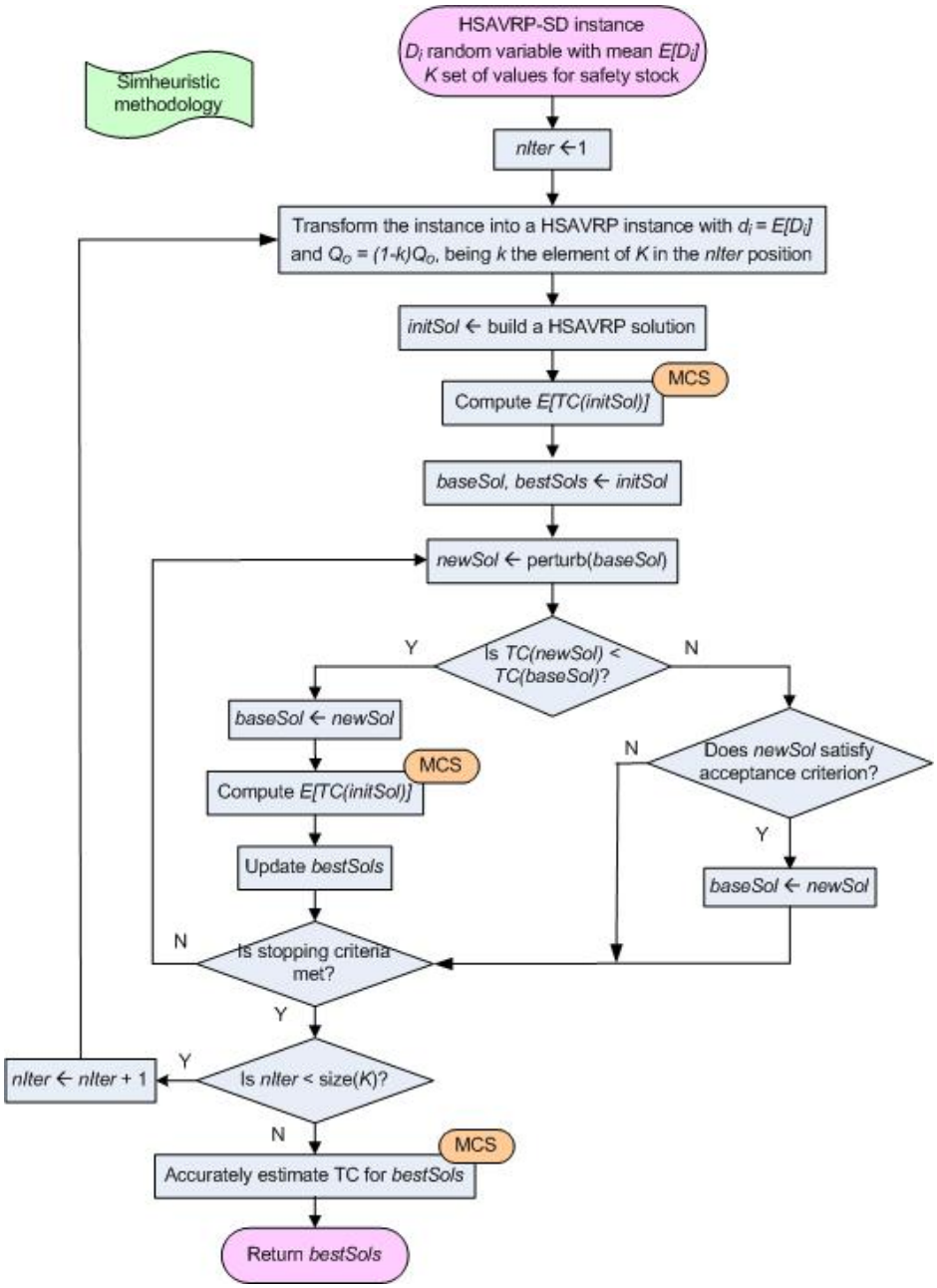


FIGURE 6.16: Flowchart of the proposed approach for the HSAVRP-SD.

The inputs are the HSAVRP-SD instance and a set K of values used to determine safety stocks. Their use leads to lower costs due to route failures (which are the costs of going from the customer being served to the depot to refill and come back to complete the delivery). However, it may also increase the number of routes needed, increasing the deterministic costs (those obtained considering that demand variances are 0). Consequently, it is required to test different values and compare expected total costs.

The algorithm starts by selecting the first value $k \in K$ and transforming the original instance into a deterministic one replacing stochastic demands by their means. Additionally, the capacities are reset to: $Q_o = (1-k)Q_o \ \ (\forall o \in F)$. The next step consists in building an initial solution (*initSol*) for the new instance and estimating the associated total costs using MCS techniques with a short number of scenarios. Afterwards, a base solution (*baseSol*) is constructed by cloning *initSol*, and a list of solutions (*bestSols*) is created, which will store the best stochastic solutions (i.e., those with the lowest expected total cost). Initially, the list includes (*initSol*). Then, a new solution (*newSol*) is obtained by perturbing *baseSol*, which involves removing a random number of routes and repairing

it. If the former has lower total costs (i.e., costs in the deterministic environment), it replaces *baseSol*, the total costs in the stochastic environment are estimated with a short MCS, and *bestSols* is updated. On the other hand, if (*newSol*) is not better than (*baseSol*), an acceptance criterion is checked to decide whether the base solution is replaced. We use a Demon-like acceptance criterion (Talbi, 2009), which allows the base solution to be deteriorated if no consecutive deteriorations take place and the degradation does not exceed the value of the last improvement. By doing this, the algorithm avoids getting stuck in a local optima. This procedure is repeated to visit different solutions until a stopping criteria is met. At this point, the algorithm is re-initialized with another value of K. When all values have been tested, the total costs of *bestSols* are accurately estimated using MCS with a larger number of scenarios. Finally, the list is returned.

Regarding the building of solutions, the SAM procedure is implemented. It can be described as follows. The procedure receives one list of customers and one of available vehicles. First, an empty global solution is created, and the list of customers is copied into a list of non-served customers. While this list is not empty, the next steps are taken. A vehicle type not used yet is selected and those customers not compatible with the selected vehicle are removed from the list. Then the problem is transformed into an homogeneous SAVRP (HoSAVRP) with no limitation on the number of vehicles that is solved with a state-of-the-art algorithm.

If the solution provided reports more routes than the number of available vehicles of the current type, some routes are discarded. This partial solution is included in the global solution. The last instructions inside the while loop update the list of available vehicles and the list of non-served customers. This process ends when all customers are assigned to a route. Finally, the global solution is returned.

The procedure for repairing solutions is exactly the same but receiving as inputs only those customers that remain to be included in a route and copying the perturbed solution into the global one when this is created.

Each HoSAVRP solution is constructed using the SR-GCWS-CS algorithm (Juan et al., 2011a). It is based on the CWS heuristic and incorporates biased randomization techniques and cache and splitting techniques, which contribute to reduce computational times. We have adapted this algorithm in order to consider asymmetric costs. For this, the easy procedure of computing savings as the mean of the two savings associated to each pair of nodes (Gruler et al., 2015b) has been applied.

6.7.2 Computational experiments

In order to test our approach, we have generalized a set of 4 classical CVRP instances from *Branch and Cut*. The same location of the nodes and demand is used. They have been modified to include the characteristics of the rich VRP.

Since we minimize a cost function, a fixed cost for using a vehicle, f_o, and a variable cost, v_o, that multiplies the distance have been established. Therefore, the cost of arc $(i, j) \in A$, $c^o_{ij} = v_o d_{ij}$, where d_{ij} is the Euclidean distance. In order to account for asymmetric costs, the cost of an edge (i, j) is incremented by 10% if the y-coordinate of the destination node j is greater than the y-coordinate of the origin node j.

An heterogeneous fleet has been proposed, with three type of vehicles. Large vehicles have a capacity equal to the one used in the benchmark, and medium and small vehicles have a reduced capacity of 75% and 50% respectively. All vehicles can serve all customers except for customers belonging to a randomly selected sub-area in which we assume that large vehicles cannot access.

Without loss of generality we have chosen the demand of a particular customer, D_i, to follow a logNormal distribution, with expected value as the demand of the benchmark instance (d_i) and variance proportional to the expected value (κd_i). The results presented next are obtained with $\kappa = 0.1$.

Several measures are computed for each solution. When a solution is evaluated with deterministic demands, the cost (Z^{det}) and the distance ($dist$) are shown. When a solution is assessed with stochastic demands, route failures may happen. Therefore, the expected cost (Z^{stoch}) and the percentage of expected route failures (r^{fail}) is displayed. Finally, the safety stock is also included.

Test cases were run on a laptop with 4 cores at 2.6GHz. Experiments were run over 5 random seeds for 60 seconds except for instance A-n80-k10 which run for 300. The name of the instances indicate the number of nodes (after the letter n). Short MCS were run for 100 scenarios, and long simulations for 10000.

6.7.3 Analysis of results

Table 6.20 compares the solution of the original CVRP instance with the current version HSAVRP with deterministic demands. When the SAM method is employed to solve the CVRP, Our Best Solution (OBS) shows to be competitive compared with the optimal (OPT) solution reported in the literature, with an average gap of 0.52%. With the solution of the HSAVRP we also report the composition of the fleet for each solution. We can observe that a mix fleet is used, motivated by the fact that some vehicles cannot access some customers. The performance of the deterministic solution is tested in the operational level with stochastic demands in Table 6.21.

A particular solution is tested under stochastic demands using MCS techniques. In Table 6.21 we can observe how the expected cost of the deterministic solution increases on average a 4% and experiences a high percentage of route failures. This is due to the fact that some routes has a filling rate of 99%. On the other hand, stochastic solutions show a filling rate more balanced among the routes, and the route failures decrease dramatically.

6.8 Conclusions

The flow of goods and products is becoming increasingly complex as a consequence of many factors such as the globalization. The weight of this sector in the gross domestic product and the employment rates of most countries require the development of inteligent algorithms to obtain efficient solutions. The constant evolution and dynamism of the sector calls for fast algorithms. Moreover, the rellevance of the social and enviromental impacts caused by this sector and the growing concern for these issues makes it necessary to study classical problems focusing on a different perspective (i.e., not analyzing only the common measures: distances or time).

While the literature on logistic transportation is extensive and varied, there are plenty of research lines to be explored. Here, both classical and novel problems have been addressed, presenting reviews, methodologies, computational experiments, and analysis of results. The main conclusions are:

- Statistical learning techniques may help to deal with uncertainty. Hybrid algorithms for routing problems allow to maximize benefits by increasing sales and total income while accounting for the distribution costs, which is a more realistic approach than the classical.
- Simheuristics are very useful to address routing problems such as the MDVRP and the WCP modeling demands as stochastic variables. Whereas solutions for the deterministic version of the problem (e.g., considering expected values to replace the random variables) tend to provide good results in scenarios characterized by a low variability, this is not true for scenarios with a higher degree of stochasticity.
- Sustainability indicators are needed to analyze the externalities of transport activities. Even if there is a high correlation between the performance of a solution in terms of distance or time, and in terms of the cost associated to other sustainability indicators, it is not perfect. As a consequence, the solutions minimizing each indicator individually may be very different in some cases.
- Smart cities require efficient and clean systems of waste coleection. There are plenty of works on this problem, most of them using real data. However, there are many lines of research, a version dealing with stochastic waste levels has been addressed.
- RVRPs encompass a large number of challenging problems with real-life applications. The HSAVRP has been tackled with a simple approach based on classical procedures but able to deal with the characteristics of the problem: heterogeneous fleet, site-dependency and asymmetric costs.

TABLE 6.20: Comparison between CVRP distance-based solutions and the HSAVRP cost-based solutions.

	Original instance (CVRP)				Deterministic HSAVRP - OBS									
Instance	Veh. capac.	OPT	OBS	Gap	Veh. used	Avail. vehicles			Z^{det}	dist	Δ dist	Used vehicles		
					L	L	M	S				L	M	S
P-n40-k5	140	458	461.7	*0.81*	5	4	2	3	2318.3	559.5	*21.18*	3	2	1
B-n41-k6	100	829	833.7	*0.56*	6	5	6	6	3656.7	1075.2	*28.97*	4	4	0
B-n45-k5	100	751	754.0	*0.39*	5	4	3	3	2907.6	802.2	*6.40*	4	2	0
A-n80-k10	100	1763	1768.7	*0.32*	10	8	6	6	6809.2	2040.9	*15.39*	6	5	0

TABLE 6.21: Table of results for the HSAVRP-SD instances.

	Deterministic HSAVRP				HSAVRP-SD					
Instance	Z^{det}	dist	Z^{sto} (1)	r^{fail}	Z^{det}	dist	Z^{sto} (2)	r^{fail}	Safety stock	Gap $_{((2)-(1))/(2)}$
P-n40-k5	2318.3	559.5	2320.7	3%	2318.3	560.5	2318.4	0%	100%	0.10%
B-n41-k6	3656.7	1075.2	4054.9	68%	3667.1	1078.6	3678.6	21%	99%	10.23%
B-n45-k5	2907.6	802.2	2972.9	56%	2912.4	804.1	2912.6	0%	99%	2.07%
A-n80-k10	6809.2	2040.9	7334.5	82%	6964.2	2063.0	7004.7	14%	98%	4.71%

Chapter 7

Application in production

This chapter studies PFSPs with stochastic processing times and a common due date. It proposes a simheuristic algorithm based on the ILS metaheuristic and Monte Carlo simulation.
It is based on the following journal article: Hatami et al. (submitted).
This work has been presented at the following conferences: Calvet et al. (2016c) and Calvet et al. (2016e).

7.1 Introduction

The manufacturing industry is facing important challenges, including fierce competitiveness, short product life cycles, increasing speed of product innovation, high product variety and quality, and rising customer expectations, among others. Industries need to find proper strategies to cope with these challenges and remain successful in the market, being one of these strategies the use of distributed manufacturing systems (Moon et al., 2002), with contrasted benefits in terms of higher product quality, lower production costs and fewer management risks (Wang, 1997; Chan et al., 2005; Kahn et al., 2013).

In distributed manufacturing systems there is an horizontal cooperation among entities when they have strategic relationships and join their individual strengths to achieve a common goal, so the complexity of manufacture is shared among different entities, resulting in conditions in which risks and costs become acceptable and market opportunities can be captured. Quite often single manufacturing centers are not able to produce products within reasonable costs and increase product diversity because of rigid organizational structures, deterministic approaches to take decisions, lack of technology and a competencies' hierarchical allocation (Sluga et al., 1998; Wang et al., 2006). As a result, single manufacturing centers are infrequent while distributed manufacturing systems are quite usual (Moon et al., 2002; Naderi and Ruiz, 2010). Constructing these collaborative manufacturing systems help industries to address market global challenges in an efficient way but their optimization is more complicated. The optimization of these systems has received a considerable attention from practitioners and the research community in recent years.

A well-established problem is the so-called distributed permutation flowshop scheduling problem (DPFSP) (Naderi and Ruiz, 2010). It consists of a set of distributed manufacturing factories with flowshop configurations. The responsibility of the factories is to produce a product composed of various jobs. Each factory has to process a certain number of jobs, and all of them should be completed at a given deadline or before. Typically, the DPFSP involves two decisions: assigning each job to be manufactured to a factory, and determining a job sequence for each factory. The classical DPFSP assumes a static environment and deterministic processing times to simplify the problem. However, real-world manufacturing systems are dynamic and often exposed to uncertainties and unforeseen events such as machine breakdown, changing due date, operator unavailability, materials out of stock, order rush, etc (Rodammer and White, 1988).

This chapter addresses a problem related to the DPFSP. It is assumed that the components have already been assigned, and the work deals with job sequencing for each flowshop. Furthermore, the processing times of the components in each flowshop are random variables. The objective is to find a robust job sequence for each factory which starts to process at the latest possible time while completes all jobs respected to the deadline. Since stochastic processing times are considered, it will only be guaranteed that jobs are finished by then with a given probability. This probability depends on the probability of each factory ending on time. Thus, if a minimum probability is required, each PFSP can not be separately solved.

The problem can be also related to the PFSP-ST. The literature on this problem is not extensive, especially when compared with the PFSP (Lin et al., 2015; Fernandez-Viagas and Framinan, 2015b; Fernandez-Viagas and Framinan, 2015c; Hsu et al., 2015), but it is becoming more popular (Baker and Altheimer, 2012; Kianfar et al., 2012; Juan et al., 2014a). Since the PFSP is an $\mathcal{NP}$-Hard problem when the number of machines are equal to or higher than 3 (Garey et al., 1976), our problem is also $\mathcal{NP}$-Hard. As a consequence, it is sensible to focus on designing heuristic or metaheuristic approaches for obtaining good solutions in reasonable CPU times.

7.2 Literature review

A review on three problems sharing characteristics with the problem analyzed is presented.

7.2.1 PFSP-ST

While the PFSP has been intensively studied during the last few decades, the PFSP-ST has received less attention. Baker and Trietsch (2011) designed heuristics for addressing the 2-machine PFSP-ST, where the processing times are independent random variables following specific probability distributions. Later, Baker and Altheimer (2012) presented a methodology for the m-machine version. In addition, several variations of the PFSP-ST have been analyzed. For instance, Allaoui et al. (2006) and Choi and Wang (2012) worked on the stochastic hybrid FSP, aiming to minimize the expected makespan. The same problem was tackled by Kianfar et al. (2012) with the goal of minimizing the average tardiness of jobs. A novel approach is applied in Zhou and Cui (2008) for tackling the multi-objective PFSP-ST, where both the flow time and delay time of jobs are minimized.

An interesting line is related to uncertainty. Basically, there are two categories: proactive (or robust) scheduling and reactive scheduling. For works falling in the first category, Roy (2010) propose constructing an original predictive schedule. The aim is to find schedules that do not require new schedules (or significant changes) when confronting disruptions. These works may consider probability distributions or sets of scenarios. Al Kattan and Maragoud (2008), Ghezail et al. (2010) and Liu et al. (2011) addressed the PFSP with uncertainty implementing proactive scheduling strategies. On the other hand, reactive scheduling consists in revising and re-optimizing schedules when unexpected events take place. A classical option is to obtain a predictive scheduling and then try to repair it according to the actual state of the system. A comprehensive review on rescheduling under disruptions is provided by Katragjini et al. (2013).

Some authors employ exact methods for addressing the PFSP-ST. A disadvantage of many of these methods is that they only work with a specific set of probability distributions and relatively small instances. Moreover, it may be difficult to adapt them for handling dependencies among processing times. Simulation techniques enable researchers to deal with these situations in a natural way. Baker and Altheimer (2012) proposed a hybrid approach combining heuristics and simulation, and tested three heuristic methods: two relying on the CDS heuristic (Campbell et al., 1970) and one on the NEH heuristic.

7.2.2 DPFSP

In this problem the jobs have not been assigned to each flowshop, so this assignment becomes part of the decision problem. This problem is also known as the distributed flowshop scheduling problem (DFSP) since Naderi and Ruiz (2010) resumed the topic for a distributed environment and makespan minimization. Nevertheless, this decision scheduling problem was first studied by David et al. (1996) based on a glass industry considering non-delay flowshops and batch production mode. Note that each factory is treated as line in this paper, but the mathematical scheduling problem inside is the same. Since then, it has been also studied under different names in the literature: *parallel flowline* (Vairaktarakis and Elhafsi, 2000) and *parallel flowshops* (Cao and Chen, 2003). Before Naderi and Ruiz (2010), the particular two-machine-flowshop layout in each factory or line has been solved using approximate algorithms by Zhang and Van De Velde (2012) and Al-Salem (2004). This particular problem turns to be a pure assignment problem due to the Johnson's rule (Johnson, 1954). For a general configuration of m machines, Naderi and Ruiz (2010) have proposed and compared several mixed integer linear programming models and constructive heuristics to solve the problem. Regarding iterated optimisation algorithm, the problem has received an

increasing attention for makespan minimization in the literature in the last years. Gao and Chen (2011) have proposed a GA using local search phases based on interchange and insertion of jobs. A TS algorithm is proposed by Gao et al. (2013). An iterated greedy (IG) algorithm without local search phases is presented by Lin et al. (2013). A SS algorithm with a reference set made up of solutions and restarts mechanisms is proposed by Naderi and Ruiz (2014). Fernandez-Viagas and Framinan (2015a) presented an IG algorithm with bounded local search phases employing properties of the problem to reduce the space of solutions. Recently, Ribas et al. (2017) have proposed several constructive heuristics and two simple iterated algorithms (IG and ILS) with variable neighbourhood searches but with zero-buffer flowshops (blocking constraint).

A particular case of the DFSP refers to the so-called *distributed assembly flowshop scheduling problem*, which combines the DFSP with assembly scheduling. In this problem, a distributed flowshop composed of f identical flowshops is followed by a single assembly operation. n jobs consisting each one of f components have to be assembled after each component has been manufactured in one of the flowshops. This decision problem includes job assignment plus the scheduling of jobs in the assembly line. The main references for this problem are Hatami et al. (2015) and Hatami et al. (2013). In the first reference, the authors consider the objective of makespan minimization, while in the second sequence-dependent setup times are assumed.

7.2.3 Assembly scheduling

This problem is also denoted *n-stage assembly* or *assembly flowshop scheduling*. In this problem m tandem lines are arranged prior to a single assembly station which is fed by the tandem lines. Using this layout, n different products (jobs) have to be manufactured, each one consisting of m components manufactured in the tandem lines. The processing time of each component in each line is different. Some authors distinguish among the *fixed* case (i.e. each component can be processed only in a given tandem line), and the *unfixed* case (i.e. each component can be processed in different factories).

For these problems, different objectives are sought, such as makespan minimization (Sung and Juhn, 2009), total flowtime (Al-Anzi and Allahverdi, 2013; Sung and Kim, 2008), due date fulfilment (Al-Anzi and Allahverdi, 2007), or the combination of several indicators (Seidgar et al., 2014). Most references refer to the 2-stage case (production followed by assembly), so they assume that each tandem line consists of a single machine. The underlying hypothesis is that there is a single processing time for each component before the assembly process. For this problem, different exact and approximate methods have been proposed, and some variants of the original problem have been tackled by Sung and Juhn (2009), where two types of components –manufactured and imported– are considered, and by Liao et al. (2015), where assembly batches are assumed. Several other variants of the problem for three stages have been addressed in the literature (see e.g. Koulamas and Kyparisis, 2001 and Komaki et al., 2017), but in none of the different versions of the problem the processing times have been assumed to be stochastic.

7.3 The DPFSP-ST

There is a set F of f distributed manufacturing factories. The shop configuration of each factory is a permutation flowshop scheduling problem (PFSP), which is a particular case of the flowshop scheduling problem (FSP) (Johnson, 1954). In the FSP, there is a set M of m machines where each job of a set N of n jobs must be processed on each machine. Each job starts to process from the first machine to the last one. Therefore, the number of operations per job is equal to the number of machines. The j^{th} operation of job i is processed on machine j, and can start if the $j-1^{th}$ operation on machine $j-1$ has been completed and machine j is free. Processing times are supposed to be known in advance and deterministic. Other classical assumptions (Baker, 1974) are: *(i)* all operations and jobs are independent and available for processing at time 0; *(ii)* all machines are continuously available and there are no breakdowns; *(iii)* each machine can process at most one job at a time; *(iv)* each job can be processed in only one machine at a time; *(v)* once an operation of a given job on a given machine has started, it cannot be interrupted (i.e., no preemption is allowed until the processing has been completed); *(vi)* setup and removal times are sequence-independent and are included in the processing times or are negligible; and *(vii)* in-process storage is considered infinite. In the FSP, there are $(n!)^m$ possible solutions since the number of job permutations per

machine is $n!$. The PFSP is a simpler version which assumes that all machines have the same job permutation and the job passing is not allowed. It has $n!$ possible solutions.

In this manufacturing layout, a product consisting of various components (jobs) has to be processed on the machines located at the factories. The processing time of each job i in each machine j, P_{ij}, is considered a random variable. The product is considered finished when all its jobs have been completed. It is required that all components are completed by a (deterministic) deadline $\tilde{d}$ with a probability not lesser than p.

Consequently, it is intended that the processing operations for job i at factory k should terminate by the deadline $\tilde{d}$. In a PFSP with a deadline, a specific job sequence has a makespan associated and the starting time can be set at the deadline minus the makespan. In contrast, the PFSP-ST is characterized by having potential different makespans under different conditions for a given job sequence. Therefore, in our setting, at least one of the three following approaches should be considered:

- To ignore the stochastic nature of the problem and replace the random variables by their representation (typically their mean). While ignoring the stochasticity may provide solutions of poor quality, it is not necessarily the case (see e.g. Framinan and Perez-Gonzalez, 2015). This is due to the fact that a deterministic optimization algorithm is faster and, as a consequence, may visit more solutions during a limited amount of time. Thus, if the level of stochasticity is low, there is a chance that solutions found are robust enough to have a good performance in a stochastic environment. This approach is labelled as *makespan* (M) in the following.

- To minimize the expected makespan. This approach stresses the average behaviour of the layout. However, if the starting time is set at the deadline minus the expected makespan, there is no guarantee that all processing operations will be completed on time. This approach is labelled as *expected makespan* (EM).

- To ensure that the final product will be finished on time with a probability p. This option allows the decision-maker to include a restriction that sets the probability of finishing on time or, conversely, the risk of a delay. This approach is labelled as *percentile makespan* (PM).

  It is assumed that the factories are independent, so p can be computed as: $p = \prod_{k=1}^{f} p_k$, and by assuming an equal allocation of probabilities we have $p_k = \sqrt[f]{p}$. Therefore, the problem is equivalent to ensure that factory k will finish its jobs with a probability p_k. In order to do so, the p_k-th makespan percentile can be computed for each factory k given a sample of makespans, and its starting time can be set to the deadline minus the makespan percentile.

Figure 7.1 shows the concepts of starting time, expected makespan and makespan percentile. The choice between the last two approaches depends on the risk-aversion of the decision-maker. For example, if the decision-maker prefers to focus on the worst outputs (i.e., the largest makespans), it is better to minimize the makespan percentile requiring a high probability. On the other hand, if she/he prefers to analyze the average case, she/he should focus on minimizing the expected makespan.

7.3.1 Methodology

Three algorithms are presented: the ILS$_M$ algorithm considers the deterministic version of the problem, while the SIM-ILS$_{EM}$ and the SIM-ILS$_{MP}$ algorithms minimize the expected makespan and the percentile makespan, respectively. For each solution returned by an algorithm, the (deterministic) makespan, the expected makespan and the makespan percentile are computed. The aim of working with different algorithms is to study and compare their behaviour. While simulation techniques are used in the SIM-ILS$_{EM}$ and the SIM-ILS$_{MP}$ algorithms, the ILS$_M$ algorithm, which works with average processing times, skips that part. From here, SIM-ILS algorithm refers to the basic structure of the SIM-ILS$_{EM}$ and the SIM-ILS$_{MP}$ algorithms.

The SIM-ILS algorithm combines the ILS metaheuristic with MCS. The metaheuristic searches for promising solutions while MCS techniques are employed to assess their performance. The promising solutions are returned by the metaheuristic when solving a (deterministic) PFSP instance, which is created from the original PFSP-ST instance by replacing the random variables P_{ij}

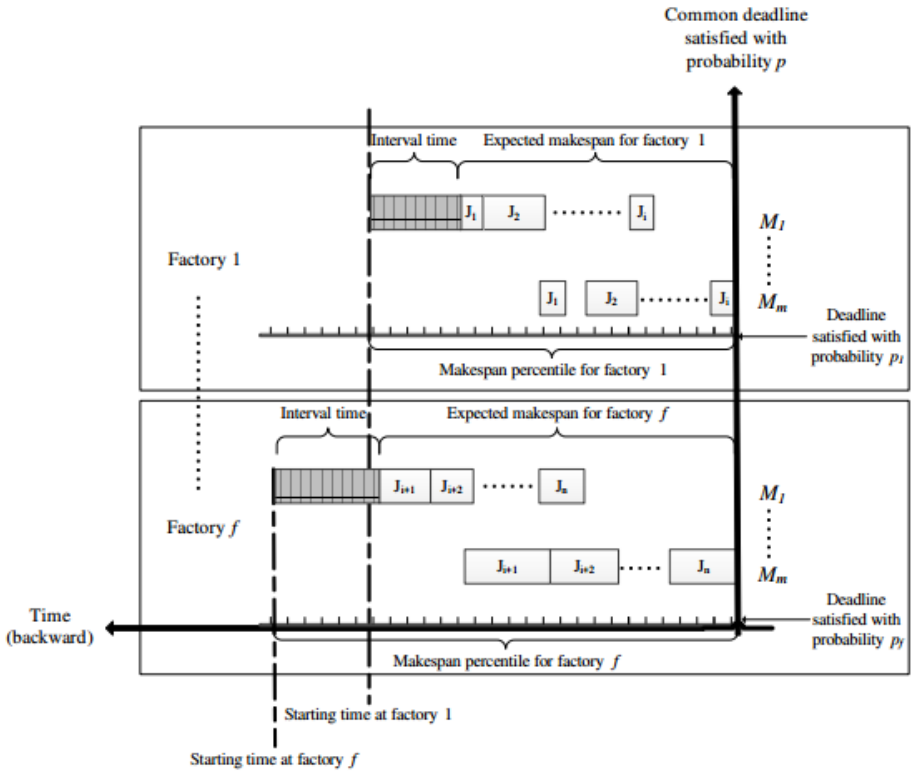


FIGURE 7.1: Starting time, expected makespan and makespan percentile in the DPFSP-ST.

by constant values p_{ij} using the means, i.e., $p_{ij} = E[P_{ij}]$. Simulation is applied to a given solution to compute the expected makespan or makespan percentile. The algorithm works with a list of best stochastic solutions found and the best deterministic solution found. The best deterministic one is the job sequence with the smallest makespan referring to the PFSP instance. Depending on the objective considered, the best stochastic solutions found are the job sequences with the smallest expected makespans or makespan percentiles, referring to the PFSP-ST instance. The algorithm starts solving the PFSP. The obtained result is set as the best deterministic solution and the best stochastic solution. During the algorithm execution, the best stochastic solutions are saved in a list with length l. This list is sorted iteratively in increasing order of the considered objective function. Thus, the solution at the first position is considered as the best stochastic solution. The steps of the algorithm are detailed in Figure 7.2 and explained below.

Generation of the initial solution

A biased-randomized version of the classical NEH heuristic (Nawaz et al., 1983) described in Juan et al. (2014a) is proposed to generate initial solutions.

Solution improvement

An iterative improvement procedure using *shift-to-left* as first-improvement type pivoting rule (Ruiz and Stützle, 2007; Juan et al., 2014a) is applied in different parts of our algorithm to improve solutions. Each iteration of the procedure consists of three steps. In the first, a position s is randomly selected, without repetition, from the current job sequence. The selected positions are saved in a selection list. In the second step, the job placed in the position s is removed from the sequence and the *shift-to-left* movement is applied, i.e., the insertion of the job in each possible position at the left side of s is tested. The makespan of each option is calculated through the accelerations of Taillard (Taillard, 1990). Finally, the job is inserted in the position resulting in the sequence with the smallest makespan. The iteration of these steps are continued until all positions have been selected or a better solution is achieved. If there is an improvement, the algorithm is restarted with an empty selection list.

Simulation

The assessment of a solution using MCS techniques follows these steps: *(1)* a number of iterations *numsim* is considered to repeat the simulation process; *(2)* a job processing time is generated

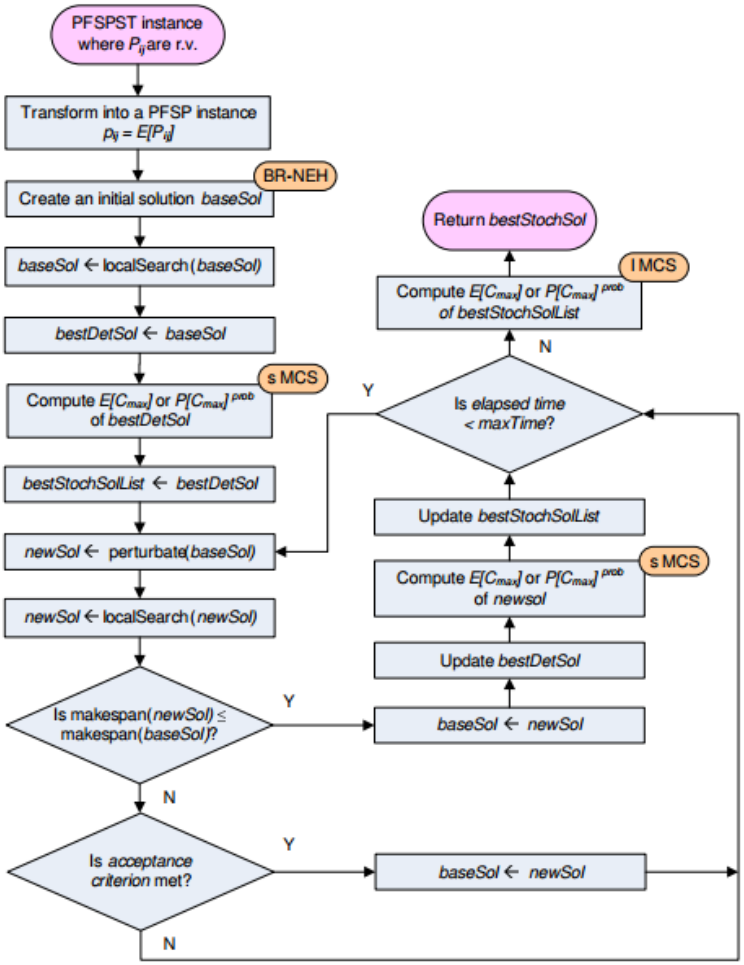


FIGURE 7.2: Flowchart of the proposed approach for the DPFSP-ST.

for each random variable according to the probability distribution associated, and the makespan is computed; *(3)* this process is repeated *numsim* times; and *(4)* a performance measure such as the expected makespan, $E[C_{max}]$, or the makespan percentile for a probability *pro*, $P[C_{max}]^{pro}$, is computed. While, the assessment of solutions during the search is done quickly (i.e., *numsim* is relatively small), a long simulation (*numsim* relatively big) is used at the end to provide accurate estimates related to the best deterministic and best stochastic solutions.

Iterated local search

A series of steps are performed iteratively during the search. Initially, a perturbation operator is applied to change the region of the current solution space and then, the new solution is improved using the local search. The simple and efficient *enhanced-swap* operator proposed by Juan et al. (2014f) is used to perturbate the solution. It takes three steps: *(1)* two different positions are selected randomly from the current job sequence; *(2)* the jobs at these positions are interchanged; and *(3)* the *shift-to-left* movement is applied for both jobs.

In the second step, the algorithm decides whether the new solution is accepted. If it has a smaller makespan than the current base solution, then the latter is replaced by the new. In this case, the best deterministic solution is accordingly updated (i.e., replaced by the new solution if this has a smaller makespan). Additionally, a short simulation is applied to check whether the best stochastic solution list has also to be updated considering the objective function value. Finally, if the new solution does not provide a smaller makespan than the current base solution, an acceptance criterion is applied. These steps are repeated until the stopping criterion based on the elapsed CPU time is reached.

Acceptance criteria

Our algorithm assigns an acceptance probability to the new solutions that are worse than the current base solution. This criterion prevents the algorithm from getting stuck in a local optima. It

is used for the first time by Hatami et al. (2015). Given a new solution π_n with a worse makespan than the current base solution π_C, the acceptance criterion decides if it is accepted or not. Let $C_{Max}(\pi_c)$ and $C_{Max}(\pi_n)$ denote the makespans of each solution. The acceptance of π_n depends on the probabilistic mechanism shown in Equation B.2, where *random* is a random number uniformly distributed between 0 and 1, and the relative percentage difference (RPD) is: $RPD = \frac{C_{(\pi_n)} - C_{(\pi_c)}}{C_{(\pi_c)}} \times 100$.

$$random \leq e^{-RPD}. \tag{7.1}$$

7.3.2 Computational experiments

The algorithms described in the previous section have been implemented as Java applications and tested on 27 instances. A standard personal computer, Intel QuadCore i5 CPU at 3.2 GHz and 4 GB RAM with Windows 7, has been used to execute all tests. This section provides the description of the instances, the tests carried out, and the numerical results. The analysis of the results is presented in the next section.

Set of instances and test

Since no benchmark instances exist for the problem analyzed, a new set is constructed based on Taillard instances (Taillard, 1993). Table B.1 gathers the following characteristics for each instance: name, total number of the jobs (total n), number of machines (m) and number of factories (f). For a given factory, each instance contains a processing time p_{ij} for job i at machine j, which describes a random variable P_{ij} following a Log-normal distribution with mean p_{ij} and variance σ_{ij}^2 set to $c \cdot p_{ij}$. In real-life applications, empirical distributions based on historical data could be used.

Table 7.1: Description of the generated instances for the DPFSP-ST.

	Total n								
	20			50			100		
f / m	5	10	20	5	10	20	5	10	20
2	Ins. 1	Ins. 4	Ins. 7	Ins. 10	Ins. 13	Ins. 16	Ins. 19	Ins. 22	Ins. 25
3	Ins. 2	Ins. 5	Ins. 8	Ins. 11	Ins. 14	Ins. 17	Ins. 20	Ins. 23	Ins. 26
4	Ins. 3	Ins. 6	Ins. 9	Ins. 12	Ins. 15	Ins. 18	Ins. 21	Ins. 24	Ins. 27

Three different levels of processing time variability c (small, medium and high) are considered and set to 0.25, 1 and 1.5, respectively. Three different values of 80%, 90% and 95% are considered for the general probability p (used only for the SIM-ILS$_{MP}$ algorithm). The maximum computational time for solving the PFSPST of each factory is limited to $0.05 \cdot n \cdot m$, which seems a reasonable amount for real-life applications. Ten seeds are randomly generated and only the best result is stored. Regarding the number of iterations for assessing solutions, 600 and 1000 runs are employed during the algorithm and at the end, respectively. Note that the selection of these values are mainly driven by the computing time available. Thus, if more time is available, then these values can be incremented in order to obtain better and more accurate results.

Results

Results are displayed in Tables 7.2-7.4, where each table represents a specific level of processing time variability: low, medium and high. Due to space limitations and the fact that results show similar trends for all three values of general probability, only those related to 90% are shown. The composition of the tables is as follows. The first column identifies the instance. The next five summarize the results of the ILS$_M$ algorithm, which considers makespan minimization. For each instance, they show the following information regarding the best solution found: $C_{max}(1)$, $E[C_{max}](2)$, $P[C_{max}]^{pro}(3)$, gap between the first two measures, computed as: $(E[C_{max}](2) - C_{max}(1))/C_{max}(1) \cdot 100$, and gap between the second and the third ones. While the first gap represents the 'extra' processing time, on average, for applying a solution assuming deterministic processing times, the second focuses on percentiles, showing the additional processing time required to finish the product with a probability of 90% (note that this time could

be negative). The next four columns provide the following results of the SIM-ILS$_{EM}$ algorithm, which minimizes the expected makespan: $E[C_{max}](4)$, $P[C_{max}]^{pro}(5)$, gap between the expected makespan of the best solutions found by the ILS$_M$ and the SIM-ILS$_{EM}$ algorithms, and the gap of percentiles among the same solutions. The third gap, which is expected to be null or negative, shows the benefit of using a simheuristic approach (i.e., taking into account the variability of the processing times) in terms of expected makespan. The fourth gap quantifies the difference of percentiles. Similarly, the next four columns refer to the best solution found by the SIM-ILS$_{MP}$ algorithm, which minimizes the makespan percentile. In particular, they contain: $E[C_{max}](6)$, $P[C_{max}]^{pro}(7)$, and gaps of expected makespans and percentiles between the best solutions found by the SIM-ILS$_{EM}$ and SIM-ILS$_{MP}$ algorithms. These gaps allow us to quantify the processing time difference based on whether we minimize one measure or the other. Finally, the last column shows the mean computational time of the three solutions obtained. In addition, a row is added at the end of each table to gather the mean gaps and computational time among instances.

Boxplots in Figure 7.3 show the distributions of gaps of $E[C_{max}]$ and $P[C_{max}]^{pro}$ regarding the best values considering the three algorithms and a probability of 90%. While we expect that the approach minimizing a given measure present a null value for the corresponding gap, this figure reveals the difference between choosing one approach or the other, allowing us to analyze the variability associated to these gaps. Focusing on the instance 14, Figure 7.4 represents the 30 solutions found (resulting of 3 algorithms and 10 seeds). Each column is a measure, and colors and line formats are used to distinguish algorithms. As the previous figure, this analysis provides insights about a “potential" trade-off between the measures. Additionally, this figure gives information about the effect of using multiple seeds.

Figure 7.5 represents the relationship between probability required, variability level of the processing times and $P[C_{max}]^{pro}$ for the instance 14 using the SIM-ILS$_{MP}$. Finally, Figure 7.6 shows the effects of different instance characteristics on $P[C_{max}]^{pro}$ considering a medium level of variability and a probability of 90%. First, an analysis of variance was carried out to identify which factors and pairwise interactions had a statistically significant effect on the results. For each of these elements (single factors or pair of them), a figure is drawn which shows the mean value associated to each level of the factor or combination of levels for pair of factors. Given the randomness in the generation of instances, we expect that all factors have significant positive effects.

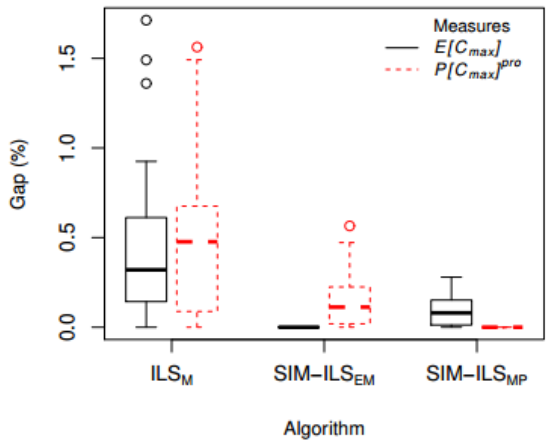


Figure 7.3: Boxplots of performance gaps for the DPFSP-ST instances considering a medium level of variability and p = 90%.

7.3.3 Analysis of results

Tables 7.2-7.4 provide detailed information on the performance of our algorithms. The following comments refer to the results of the ILS$_M$ algorithm. Mean gaps between C_{max} and $E[C_{max}]$ for small, medium and high levels of variability are 1.02%, 2.78%, and 3.74%, respectively. These values between $E[C_{max}]$ and $P[C_{max}]^{pro}$ are 1.99%, 3.88%, and 4.73%. These values quantify the extra processing time required, on average, when variability is not considered, and the processing time needed to satisfy the deadline with a probability of 90%. For example, in the scenario

TABLE 7.2: Results considering low level of variability (c = 0.25) and general probability p = 90%.

		ILS$_M$				SIM-ILS$_{EM}$				SIM-ILS$_{MP}$				
Instance	C_{max} (1)	$E[C_{max}]$ (2)	$P[C_{max}]^{pro}$ (3)	Gap (2-1) (%)	Gap (3-2) (%)	$E[C_{max}]$ (4)	$P[C_{max}]^{pro}$ (5)	Gap (4-2) (%)	Gap (5-3) (%)	$E[C_{max}]$ (6)	$P[C_{max}]^{pro}$ (7)	Gap (6-4) (%)	Gap (7-5) (%)	Mean CPU time
1	1505	1516.00	1555.54	0.73	2.61	1512.33	1552.96	-0.24	-0.17	1514.19	1551.75	0.12	-0.08	5.00
2	1758	1772.30	1829.89	0.81	3.25	1762.47	1824.72	-0.55	-0.28	1765.96	1822.13	0.20	-0.14	3.99
3	1862	1869.94	1949.62	0.43	4.26	1864.56	1944.33	-0.29	-0.27	1865.41	1942.73	0.05	-0.08	4.50
4	2154	2169.84	2216.99	0.74	2.17	2167.72	2213.02	-0.10	-0.18	2169.41	2212.05	0.08	-0.04	9.99
5	2829	2854.76	2927.28	0.91	2.54	2852.35	2923.38	-0.08	-0.13	2854.23	2921.13	0.07	-0.08	10.00
6	3179	3194.56	3295.33	0.49	3.15	3189.44	3294.94	-0.16	-0.01	3191.61	3288.55	0.07	-0.19	7.99
7	3413	3451.80	3503.35	1.14	1.49	3440.75	3495.76	-0.32	-0.22	3442.72	3493.16	0.06	-0.07	19.99
8	4306	4333.25	4421.54	0.63	2.04	4332.04	4424.23	-0.03	0.06	4334.49	4418.29	0.06	-0.13	20.00
9	5733	5760.93	5904.21	0.49	2.49	5752.36	5890.05	-0.15	-0.24	5756.43	5884.47	0.07	-0.09	14.00
10	2960	2987.24	3040.81	0.92	1.79	2960.68	3020.68	-0.89	-0.66	2964.55	3019.89	0.13	-0.03	12.46
11	3250	3285.55	3352.58	1.09	2.04	3272.02	3349.40	-0.41	-0.09	3273.94	3346.25	0.06	-0.09	12.49
12	3378	3417.10	3505.31	1.16	2.58	3392.12	3501.08	-0.73	-0.12	3395.54	3490.67	0.10	-0.30	12.49
13	3572	3620.75	3669.11	1.36	1.34	3620.75	3669.11	0.00	0.00	3620.75	3669.11	0.00	0.00	25.01
14	4031	4093.01	4172.92	1.54	1.95	4083.66	4163.44	-0.23	-0.23	4084.43	4156.70	0.02	-0.16	25.00
15	4574	4620.28	4728.14	1.01	2.33	4605.65	4720.76	-0.32	-0.16	4609.00	4714.98	0.07	-0.12	24.99
16	5058	5132.22	5194.14	1.47	1.21	5131.72	5191.09	-0.01	-0.06	5131.72	5191.09	0.00	0.00	49.99
17	6039	6113.22	6205.43	1.23	1.51	6109.34	6208.90	-0.06	0.06	6113.31	6204.41	0.06	-0.07	49.97
18	7013	7090.64	7215.40	1.11	1.76	7081.97	7215.77	-0.12	0.01	7085.22	7204.73	0.05	-0.15	49.98
19	5797	5840.48	5913.77	0.75	1.25	5810.07	5891.05	-0.52	-0.38	5818.16	5889.75	0.14	-0.02	24.98
20	5819	5854.95	5967.56	0.62	1.92	5825.96	5939.00	-0.50	-0.48	5832.76	5935.59	0.12	-0.06	24.95
21	6065	6104.18	6236.87	0.65	2.17	6078.89	6220.23	-0.41	-0.27	6083.24	6217.93	0.07	-0.04	24.98
22	6235	6324.25	6392.90	1.43	1.09	6324.25	6392.90	0.00	0.00	6326.47	6388.82	0.04	-0.06	49.98
23	6414	6502.24	6598.61	1.38	1.48	6498.90	6592.36	-0.05	-0.09	6498.90	6592.36	0.00	0.00	50.01
24	7183	7289.80	7422.74	1.49	1.82	7283.76	7418.55	-0.08	-0.06	7283.75	7407.78	0.00	-0.15	50.00
25	7603	7697.50	7763.14	1.24	0.85	7697.50	7763.14	0.00	0.00	7697.50	7763.14	0.00	0.00	100.03
26	8598	8710.46	8823.77	1.31	1.30	8710.46	8823.77	0.00	0.00	8710.46	8823.77	0.00	0.00	100.01
27	9892	10038.66	10178.40	1.48	1.39	10029.92	10175.82	-0.09	-0.03	10029.92	10175.82	0.00	0.00	99.99
Mean				1.02	1.99			-0.24	-0.15			0.06	-0.08	32.69

Table 7.3: Results considering medium level of variability (c = 1) and general probability p = 90%.

	ILS$_M$					SIM-ILS$_{EM}$				SIM-ILS$_{MP}$				
Instance	C_{max} (1)	$E[C_{max}]$ (2)	$P[C_{max}]^{pro}$ (3)	Gap (2-1) (%)	Gap (3-2) (%)	$E[C_{max}]$ (4)	$P[C_{max}]^{pro}$ (5)	Gap (4-2) (%)	Gap (5-3) (%)	$E[C_{max}]$ (6)	$P[C_{max}]^{pro}$ (7)	Gap (6-4) (%)	Gap (7-5) (%)	Mean CPU time
1	1505	1541.15	1617.25	2.40	4.94	1533.01	1614.37	-0.53	-0.18	1533.96	1606.78	0.06	-0.47	5.00
2	1758	1799.07	1927.61	2.34	7.14	1782.58	1901.69	-0.92	-1.34	1787.49	1897.95	0.28	-0.20	3.99
3	1862	1880.42	2046.20	0.99	8.82	1876.25	2041.74	-0.22	-0.22	1879.09	2030.26	0.15	-0.56	4.50
4	2154	2208.99	2298.54	2.55	4.05	2203.58	2295.21	-0.25	-0.14	2205.94	2287.00	0.11	-0.36	10.00
5	2829	2898.36	3035.07	2.45	4.72	2896.75	3033.22	-0.06	-0.06	2899.76	3032.76	0.10	-0.02	9.99
6	3179	3223.10	3424.39	1.39	6.25	3215.97	3414.35	-0.22	-0.29	3215.97	3414.35	0.00	0.00	8.00
7	3413	3517.97	3617.33	3.08	2.82	3493.77	3603.77	-0.69	-0.37	3495.26	3594.01	0.04	-0.27	19.99
8	4306	4387.69	4552.74	1.90	3.76	4380.10	4564.70	-0.17	0.26	4382.97	4552.50	0.07	-0.27	19.99
9	5733	5811.38	6094.35	1.37	4.87	5792.84	6067.04	-0.32	-0.45	5797.46	6055.08	0.08	-0.20	14.00
10	2960	3030.64	3136.25	2.39	3.48	2979.60	3094.51	-1.68	-1.33	2983.69	3090.15	0.14	-0.14	12.49
11	3250	3336.39	3502.73	2.66	4.99	3321.23	3475.18	-0.45	-0.79	3326.65	3466.45	0.16	-0.25	12.49
12	3378	3482.60	3673.45	3.10	5.48	3431.44	3638.61	-1.47	-0.95	3435.06	3627.96	0.11	-0.29	12.51
13	3573	3710.11	3810.50	3.84	2.71	3708.59	3810.36	-0.04	0.00	3708.59	3810.36	0.00	0.00	25.00
14	4031	4181.10	4332.06	3.72	3.61	4162.39	4322.68	-0.45	-0.22	4174.02	4317.41	0.28	-0.12	25.01
15	4574	4704.41	4911.53	2.85	4.40	4683.63	4901.01	-0.44	-0.21	4686.14	4896.47	0.05	-0.09	25.00
16	5058	5249.31	5356.33	3.78	2.04	5243.28	5349.75	-0.11	-0.12	5243.28	5349.75	0.00	0.00	50.00
17	6039	6241.34	6419.70	3.35	2.86	6226.07	6409.99	-0.24	-0.15	6229.50	6405.71	0.05	-0.07	49.98
18	7013	7205.94	7440.63	2.75	3.26	7205.49	7448.99	-0.01	0.11	7205.94	7440.63	0.01	-0.11	49.98
19	5797	5897.76	6042.62	1.74	2.46	5855.19	6010.80	-0.72	-0.53	5866.03	6001.97	0.19	-0.15	24.96
20	5819	5954.39	6150.09	2.33	3.29	5874.48	6086.12	-1.34	-1.04	5887.17	6081.12	0.22	-0.08	24.92
21	6065	6181.98	6426.37	1.93	3.95	6128.44	6403.68	-0.87	-0.35	6144.92	6395.87	0.27	-0.12	24.99
22	6233	6449.74	6575.00	3.48	1.94	6441.78	6569.46	-0.12	-0.08	6441.78	6569.46	0.00	0.00	50.00
23	6415	6663.16	6858.39	3.87	2.93	6633.10	6821.64	-0.45	-0.54	6642.01	6820.06	0.13	-0.02	49.99
24	7183	7464.61	7726.46	3.92	3.51	7428.87	7676.50	-0.48	-0.65	7440.15	7674.73	0.15	-0.02	49.98
25	7600	7863.12	7990.74	3.46	1.62	7863.12	7990.74	0.00	0.00	7863.12	7990.74	0.00	0.00	100.00
26	8592	8897.85	9103.21	3.56	2.31	8897.85	9103.21	0.00	0.00	8897.85	9103.21	0.00	0.00	99.98
27	9888	10267.27	10539.56	3.84	2.65	10250.60	10535.72	-0.16	-0.04	10252.56	10529.88	0.02	-0.06	99.97
Mean				2.78	3.88			-0.46	-0.36			0.10	-0.14	32.69

TABLE 7.4: Results considering high level of variability (c = 1.5) and general probability p = 90%.

	ILS$_M$					SIM-ILS$_{EM}$				SIM-ILS$_{MP}$				
Instance	C_{max} (1)	$E[C_{max}]$ (2)	$P[C_{max}]^{pro}$ (3)	Gap (2-1) (%)	Gap (3-2) (%)	$E[C_{max}]$ (4)	$P[C_{max}]^{pro}$ (5)	Gap (4-2) (%)	Gap (5-3) (%)	$E[C_{max}]$ (6)	$P[C_{max}]^{pro}$ (7)	Gap (6-4) (%)	Gap (7-5) (%)	Mean CPU time
1	1505	1558.65	1652.71	3.56	6.03	1543.93	1645.57	-0.94	-0.43	1547.66	1634.96	0.24	-0.64	5.00
2	1758	1815.86	1971.38	3.29	8.56	1793.26	1936.85	-1.24	-1.75	1796.58	1932.27	0.19	-0.24	3.99
3	1862	1899.64	2088.00	2.02	9.92	1883.39	2087.17	-0.86	-0.04	1887.72	2070.03	0.23	-0.82	4.50
4	2154	2230.49	2338.89	3.55	4.86	2223.55	2331.53	-0.31	-0.31	2225.38	2325.86	0.08	-0.24	10.00
5	2829	2923.50	3100.37	3.34	6.05	2920.98	3095.04	-0.09	-0.17	2925.05	3089.05	0.14	-0.19	9.99
6	3179	3250.82	3500.13	2.26	7.67	3231.39	3488.72	-0.60	-0.33	3239.13	3474.39	0.24	-0.41	8.00
7	3413	3550.25	3693.79	4.02	4.04	3521.54	3653.49	-0.81	-1.09	3522.88	3647.18	0.04	-0.17	20.00
8	4306	4417.80	4645.59	2.60	5.16	4408.96	4633.95	-0.20	-0.25	4414.71	4618.62	0.13	-0.33	19.99
9	5733	5829.65	6148.97	1.69	5.48	5815.95	6148.41	-0.23	-0.01	5820.54	6141.05	0.08	-0.12	14.00
10	2960	3051.56	3186.60	3.09	4.43	2989.60	3128.79	-2.03	-1.81	2989.60	3128.79	0.00	0.00	12.46
11	3250	3369.09	3547.77	3.66	5.30	3348.04	3530.25	-0.62	-0.49	3354.05	3526.75	0.18	-0.10	12.50
12	3378	3510.02	3741.10	3.91	6.58	3455.00	3711.78	-1.57	-0.78	3459.46	3691.02	0.13	-0.56	12.49
13	3570	3745.06	3862.43	4.90	3.13	3745.06	3862.43	0.00	0.00	3745.06	3862.43	0.00	0.00	24.99
14	4031	4221.27	4401.91	4.72	4.28	4206.78	4396.52	-0.34	-0.12	4210.35	4395.90	0.08	-0.01	24.99
15	4574	4743.71	4997.22	3.71	5.34	4729.26	5001.05	-0.30	0.08	4737.43	4987.14	0.17	-0.28	24.99
16	5058	5312.82	5448.19	5.04	2.55	5304.92	5443.07	-0.15	-0.09	5304.92	5443.07	0.00	0.00	50.01
17	6039	6301.25	6533.04	4.34	3.68	6289.15	6517.20	-0.19	-0.24	6293.81	6507.38	0.07	-0.15	49.97
18	7013	7282.57	7597.07	3.84	4.32	7273.14	7594.89	-0.13	-0.03	7277.18	7565.94	0.06	-0.38	49.99
19	5797	5967.38	6140.62	2.94	2.90	5881.63	6066.46	-1.44	-1.21	5889.21	6054.73	0.13	-0.19	24.99
20	5819	5990.94	6236.24	2.95	4.09	5908.93	6161.28	-1.37	-1.20	5916.77	6155.52	0.13	-0.09	24.94
21	6065	6230.30	6563.94	2.73	5.36	6169.34	6505.67	-0.98	-0.89	6172.18	6485.97	0.05	-0.30	24.98
22	6237	6531.98	6675.40	4.73	2.20	6509.74	6672.00	-0.34	-0.05	6510.28	6662.21	0.01	-0.15	49.99
23	6411	6718.36	6953.37	4.79	3.50	6707.55	6922.70	-0.16	-0.44	6707.55	6922.70	0.00	0.00	50.00
24	7183	7519.80	7820.43	4.69	4.00	7499.62	7802.18	-0.27	-0.23	7499.62	7802.18	0.00	0.00	50.00
25	7603	7970.91	8150.20	4.84	2.25	7962.01	8126.37	-0.11	-0.29	7962.01	8126.37	0.00	0.00	100.00
26	8600	9011.27	9264.51	4.78	2.81	9011.27	9264.51	0.00	0.00	9011.27	9264.51	0.00	0.00	99.98
27	9892	10386.80	10731.39	5.00	3.32	10354.76	10707.00	-0.31	-0.23	10354.76	10707.00	0.00	0.00	99.99
Mean				3.74	4.73			-0.58	-0.46			0.09	-0.20	32.69

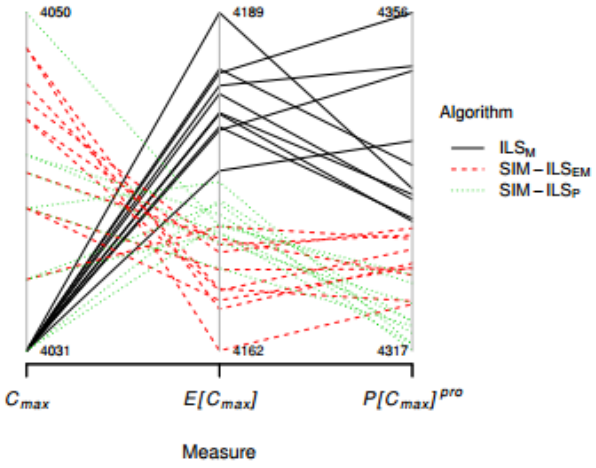


FIGURE 7.4: Parallel coordinates plot showing different measures for the DPFSP-ST instance '14', considering a medium level of variability and 10 seeds.

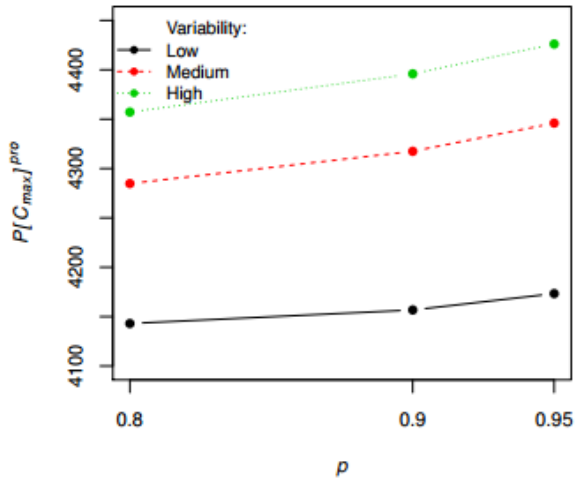


FIGURE 7.5: $P[SC_{max}]^{pro}$ as function of general probability and variability level for the DPFSP-ST instance '14' considering the SIM-ILS$_{MP}$ algorithm.

of low variability, the processing time will be on average 1.02% higher than that assumed, and the processing time needed to finish with the specific probability will be 1.99% higher than the $E[C_{max}]$. Both gaps increase as the variability is incremented. Comparing the results of the ILS$_M$ and the SIM-ILS$_{EM}$ algorithms, the mean gaps of $E[C_{max}]$ (−0.24%, −0.46% and −0.58%) and $P[SC_{max}]^{pro}$ (−0.15%, −0.36% and −0.46%) quantify the benefits of using a simheuristic algorithm. Regarding the results of the SIM-ILS$_{EM}$ and SIM-ILS$_{MP}$ algorithms, the mean gaps of $E[C_{max}]$ (0.06%, 0.10% and 0.09%) and $P[C_{max}]^{pro}$ (−0.08%, −0.14% and −0.20%) at different level of variability, evidence the benefits of using one or the other approach. Thus, the main findings are: *(i)* ignoring the variability in processing times may have an important effect on the performance measures (even in scenarios with a low level of variability); *(ii)* the solutions found by the SIM-ILS$_{EM}$ and the SIM-ILS$_{MP}$ algorithms are relatively similar in terms of these measures but not equal; and *(iii)* the gaps tend to increase with the variability, i.e., minimizing the expected makespan is almost equivalent to consider the makespan percentile when the variability is low, but the difference increases as the variability is incremented. As a consequence, a decision-maker have to assess whether he prefers to minimize the expected makespan (i.e., processing finished at the deadline, on average) or the percentile (i.e., be sure that the processing will be finished at the deadline or before with a given probability), which may be seen as a more conservative or risk-aversion approach.

Figure 7.3 compares performance measures among the algorithms proposed. The distributions

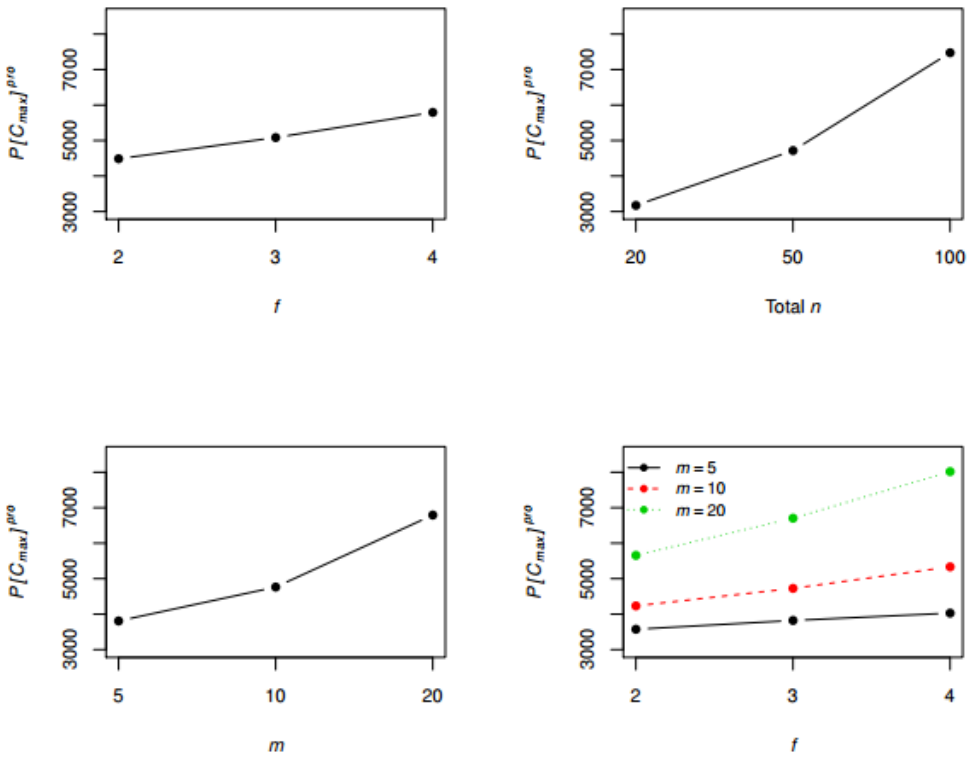


FIGURE 7.6: Effect of different DPFSP-ST instance characteristics on $P[C_{max}]^{pro}$ considering the SIM-ILS$_{MP}$ algorithm.

of the gaps are relatively symmetric, with few outliers on right tails. It is easy to see that the biggest gaps are related to the ILS$_M$ algorithm, while the gaps referring to $P[SC_{max}]^{pro}$ values are higher. Similarly, Figure 7.4 shows that there is a stronger correlation between the simheuristic-based algorithms in the sense that the profiles are similar. It is interesting to analyze the differences among the solutions obtained with multiple seeds. For instance, while solutions of the ILS$_M$ algorithm have the same or a similar C_{max}, these solutions may differ significantly in the other measures (i.e., there are solutions more robust than others). For the instance studied, the ranges of the last two measures are higher than that of the first.

Figure 7.5 represents a valuable tool for a decision-maker. It analyzes the relationship between the probability required to process a product at the deadline or before and the processing time needed. As the probability tends to 1 (i.e., no risk) the processing time tends to infinite. Instead of having a single solution, the decision-maker may choose the best option (given risk-aversion, company policies/situation, etc.) among many. As expected, for a given p value, $P[C_{max}]^{pro}$ increases as the variability is incremented.

Figure 7.6 reveals that factors total n, m, f, and the interaction between f and m have statistically significant and positive effects (when considering the others elements) on $P[C_{max}]^{pro}$. The ranges related to total n and m are the highest. While the effects of f and total n seem lineal, the effect of m draws a convex function. Focusing on the interaction, it can be concluded that the effect of f is positive for any value of m, but $P[C_{max}]^{pro}$ increases as m is incremented.

7.4 Conclusions

The manufacturing industry is becoming increasingly complex and competitive. Companies need powerful optimization algorithms to design proper strategies that make them efficient in order to remain in the market. Although there is an extensive literature on classical scheduling problems, there is a lack of works on richer and more realistic problems. This chapter studies a novel problem called DPFSP-ST. It consists in the manufacturing of a product that requires several jobs that are performed in independent factories. The sub-problem of each factory can be modelled as a PFSP-ST. All factories are expected to finish at a given deadline or before. This problem describes

several real-life applications where a company acquires intermediate products from others and assembles them to obtain a final product with a higher added value.

Three algorithms are proposed to deal with this problem which aim to minimize a different objective function: the makespan (ignoring the stochasticity), the expected makespan and the makespan percentile given a probability p. This percentile is the value below which a given proportion p of makespans fall when simulating scenarios, and can be interpreted as follow: if the starting time in a factory is set to the deadline minus this percentile, the processing of the product will be finished before or at the deadline with a probability p. While all algorithms rely on the ILS metaheuristic, the second and the third ones are simheuristic algorithms, i.e., integrate MCS techniques in order to deal with the stochasticity. Note that the second algorithm is intended to provide good results on average whereas the third one aims to guarantee that the manufacturing will be finished before or at the deadline with a given probability. A set of computational experiments allow us to compare the algorithms in terms of makespan, expected makespan and makespan percentile, and quantify these differences. It is proven that: *(i)* gaps among algorithms for each measure increase as the level of stochasticity is incremented; *(ii)* while there is a strong correlation between simheuristic algorithms (in the sense that solutions having the best performance in terms of expected makespan are also of good quality regarding makespan percentile, and the other way around), it is weaker between the first algorithm and any of the others; *(iii)* in some cases the differences between the second and the third algorithm may be significant, so a priority must be set by the decision-maker; *(iv)* the fact that the algorithms are so fast enable the running of the third one considering different probabilities, which provides a deeper insight of the relationship between probability (related to the risk-aversion, i.e., how sure decision-maker wants to be about finishing at a given deadline or before) and makespan percentile (i.e., how much time he needs to start before the deadline); *(v)* the effect of using different seeds is significant; and *(vi)* the makespan percentile linearly depends on the number of factories, jobs and machines, and the interaction between number of factories and machines.

Chapter 8

Applications in finance

This chapter reviews works using metaheuristics in portfolio optimization and risk management, studies the deterministic and stochastic portfolio optimization problem, and presents an application to stocks and individual commodity futures contracts. It proposes hybrid algorithms based on the ILS or the VNS metaheuristics, and MCS.
It is based on the following journal articles: Doering et al. (submitted[a]), Kizys et al. (submitted), Calvet et al. (submitted[b]), and Doering et al. (submitted[b]).
This work has been presented at the following conferences: Doering et al. (2016b), Calvet et al. (2016f), and Doering et al. (2016a).

8.1 Introduction

Investments play an essential role in improvements of welfare standards. This striving for improvement is represented through the formulation of optimization problems for most of the questions in financial economics. Traditionally, exact methods have been employed. The current internationalization and integration of financial markets and institutions has caused financial decision-making processes to become even more complex, both in terms of associated constraints as well as in terms of the instances to solve. Metaheuristics constitute an attractive alternative for problem solving in several knowledge areas in which real-time decisions are required. Applications of metaheuristics in the financial sector are presented in Gilli et al. (2011).

The first section reviews the literature on metaheuristic optimization applications for portfolio and risk management in a systematic way. The chapter identifies the linkages between portfolio optimization and risk management. It is expected that the revocation of the strict classification of financial COPs can lead to a methodological transfer of knowledge. In addition, the trends that have gradually become apparent in the literature and are expected to dominate future research in this knowledge area are outlined.

The second section focuses on a single-period version of the so-called constrained mean-variance POP. Three realistic constraints are considered. First, justified on the grounds of the investor's preference and/or taste, the *pre-assignments* force some specific assets to be included in the portfolio. Second, the *quantity* constraint keeps the quantity of each selected asset within user-specified *floor* and *ceiling* values. The ceiling rules out excessive exposure to a specific asset. The floor is introduced in order to rule out the possibility of tiny (and therefore disproportionately costly) fractions of assets to be included in the portfolio. Third, the *cardinality* constraint, which imposes a floor and a ceiling on the number of assets included in the portfolio, accounts for the fact that diversification benefits decrease when the portfolio features a huge number of assets. In the presence of these rich constraints, the problem becomes $\mathcal{NP}$-hard (Bienstock, 1996) and, thus, exact optimization methods quickly lose their efficiency as the number of considered assets grows. The cardinality constraint also implies that the mean-variance frontier can become discontinuous for certain values of expected return (Chang et al., 2000). Because our research involves the constrained efficient frontier (CEF), it is devised a matheuristic algorithm for rich portfolio optimization (ARPO) that is based on the combination of an ILS metaheuristic, quadratic programming, and biased randomization strategies.

Contrary to the well-established real-life constraints, the growing body of literature assumes constant rates of returns and covariances. This empirically unsupported assumption poses a key limitation when real-life approaches are sought. The aim of the third section is to address this limitation. Indeed, since asset returns are random variables that obey certain probability density

functions, and future returns are unpredictable, the minimum desired rate of return may not be attained with certainty. More concretely, we relax the above simplifying assumptions and consider rates of returns and covariances as random variables. The resulting problem is known as the SPOP. Here a simheuristic algorithm to solve the SPOP is proposed based on the VNS metaheuristic. While the metaheuristic generates promising portfolios for a deterministic version of the problem -the one obtained when expected values are considered-, simulation techniques are applied to: *(i)* estimate the expected risk of these portfolios under uncertainty conditions; *(ii)* complete a risk analysis on each portfolio; and *(iii)* provide feedback to the metaheuristic in order to better guide the searching process.

Finally, the last section addresses the rich POP, considering individual futures contracts in addition to stocks. Recently, stock markets have become more integrated, resulting in higher positive correlation among individual stocks and thus diminishing successful diversification (You and Daigler, 2012). Because most research on metaheuristics applied to POPs relies on pre-established benchmarks, the outcomes of such a development on the quality of the established portfolios cannot be detected. Thus, it would be convenient to include a second asset class to exemplify possible diversification benefits. Individual commodity futures contracts are selected because they have been found to have low correlations with stocks (Jensen et al., 2002; Chong and Miffre, 2010). Their correlational properties have been found to be caused among others by an opposite reaction of futures to macroeconomic shocks (Silvennoinen and Thorp, 2013; Bansal et al., 2014).

8.2 Survey on metaheuristics in portfolio optimization and risk management

The increasing popularity of the application of metaheuristics to POPs and risk management problems (RMPs) is depicted in Figure 8.1 based on Scopus-indexed publications. The search for POPs was conducted by examining the articles that explicitly consider portfolio optimization, index tracking or project selection in the abstract, title or keywords and make use of metaheuristics. For risk management problems, the search terms were bankruptcy, credit risk or stock or foreign exchange trading. In the case of portfolio optimization, it becomes obvious that the trend in publications is increasing. Continuing increases in computing power, the advancement of metaheuristic frameworks and parallelization strategies favour metaheuristics when dealing with NP-hard financial COPs. On the contrary, risk management problems seem to have received much less attention. These proportions are broken down in Figure 8.2, which shows that traditional portfolio optimization represents the majority of metaheuristic applications.

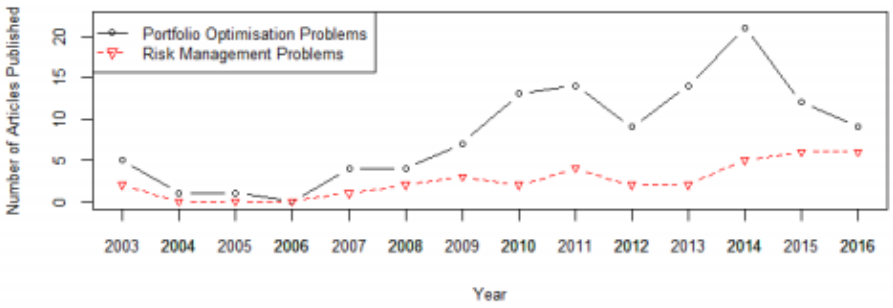


FIGURE 8.1: Scopus-indexed publications applying metaheuristics to POPs and RMPs for the period 2003 to 2016-1 (first semester).

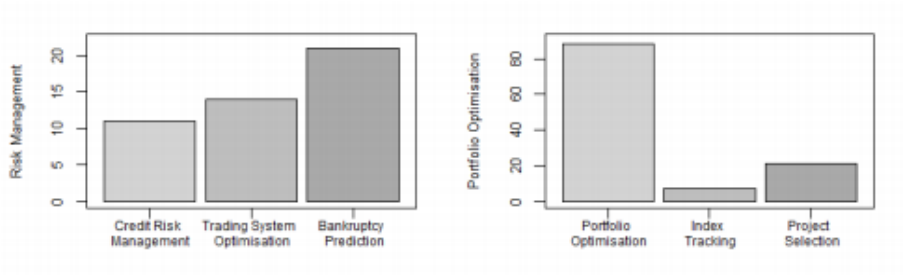


FIGURE 8.2: Number of publications on POPs and RMPs.

8.2.1 Portfolio optimization

Since Markowitz (1952) developed the portfolio optimization theory centred around the mean-variance approach, the academic community has been highly engaged in advancing the tools for portfolio optimization. The theory is based on two constituting assumptions, namely: *(i)* the financial investors being concerned with the expected returns; and *(ii)* the risk of their respective investment. It is thus the goal to minimize the level of risk expressed through the portfolio variance for a given expected return level, resulting in the so-called unconstrained efficient frontier, from which the portfolio choice is determined by the risk awareness of the investor. This established the POP, which is a strategy of: *(i)* selection of financial assets; and *(ii)* determination of the optimal weights allocated to those assets that results in a desired portfolio return and associated minimum level of risk. Based on the investor's involvement with the asset selection, two types of investment management strategies can be identified. On the one hand, active investment strategies aim at beating market returns. On the other hand, passive investment strategies aim at replicating a benchmark index. This strategy has become specifically popular with equity funds and although it is originally based on the efficient market hypothesis, passively indexed funds can still outperform active funds and have shown to do so on average due to the increased management costs of active funds in the presence of market failures (Malkiel, 2003). According to these conclusions, index replication is not solely a hedging strategy, but provides stable profitability.

Table 8.1 presents a summary of the metaheuristics applied to each of the problems reviewed: single-objective portfolio optimization, multi-objective portfolio optimization, index tracking, enhanced index tracking, and project portfolio selection. The number of articles found on each topic and metaheuristic is included inside each cell. The classical portfolio optimization is an active investment strategy, particularly when active re-balancing of the portfolio takes place in multi-period observations and, by its nature, investment appraisal requires the active selection of project portfolios. Index tracking is traditionally a passive strategy, while enhanced index tracking involves active management to some extent.

TABLE 8.1: Application of metaheuristics and hybridization to POPs.

Optimization problem	Single-solution search				Population-based search										Hybrid
	SA	TS	FD	SD	GA	FA	ACO	DE	EA	ABC	PSO	IWO	AIS	SS	
Single-objective portfolio optimization	2	3	1	1	2				1		3				3
Multi-objective portfolio optimization					2	2		2	1	1	4				2
Index tracking	1				2			2	3			1			3
Enhanced index tracking	1	1	1	1	2			1			1		2		1
Project selection		3			2		6				1			2	5

Traditional portfolio optimization

While the original Markowitz problem can be solved using quadratic programming, metaheuristics have increasingly been employed to cope with the fact that the problem becomes NP-hard when more realistic constraints are introduced (Beasley, 2013). In effect, cardinality constraints, quantity constraints, and pre-assignment constraints have received overwhelming attention in the literature. The cardinality constraint defines a lower and upper limit for the numbers of assets included in the portfolio. While the lower bound aims at portfolio diversification, the upper bound accounts for the fact that marginal benefits of diversification diminish after a certain threshold (Maringer, 2005), which increases managerial efforts and transaction costs. The quantity constraint sets boundaries for the weights of included assets. While the lower limit ensures a minimum

investment as smaller investments may be prohibitively costly due to transaction costs (Kolm et al., 2014), the upper limit prevents excessive exposure to a particular asset. Finally, the pre-assignment constraint enables the investor to include certain assets in the portfolio based on individual preferences independent from their risk-return characteristics.

Single-objective portfolio optimization

The classical POP can be considered a single-objective optimization problem with either one of the following model formulations: the investor minimizes the risk exposure subject to a minimum attainable expected return, or the investor maximizes the expected return for a given maximum level of risk. The first variant can be formulated as follows (Chang et al., 2000): A quadratic objective function is computed by aggregating over the covariances of the constituent asset returns and then minimized:

$$Min \sum_{i=1}^{N} \sum_{j=1}^{N} w_i w_j \sigma_{ij}, \tag{8.1}$$

subject to a minimum desired rate of return, the constraint that the weights have to add up to one, and the constraint that all asset weights must lie between zero and one, inclusive, thus eliminating short selling as a measure of preventing investors from excessive risk-taking by restricting them to the available budget. In formal terms:

$$\sum_{i=1}^{N} w_i \mu_i \geq R, \tag{8.2}$$

$$0 \leq w_i \leq 1, \quad \forall i \in \{1, 2, \ldots, N\} \tag{8.3}$$

where N is the total number of available assets, μ_i is the expected return of an asset i, R is the minimum required return, w are the respective weights of the assets making up the portfolio, and σ_{ij} is the covariance between two assets i and j.

Chang et al. (2000) solved it using three metaheuristic approaches (GA, SA, and TS) in order to generate a cardinality-constrained efficient frontier. They suggested pooling the results from the different approaches because no single heuristic was uniformly dominating in all observed datasets. However, Soleimani et al. (2009) introduced sector capitalization and minimum transaction lots as further constraints and found that the GA they developed outperformed TS and SA. Following the suggestion of Chang et al. (2000) and combining GA, TS, and SA, Woodside-Oriakhi et al. (2011) explored the pooling option. They found that, on average, SA contributes little to the performance of the process and that thus a pooled GA and TS algorithm is superior to single metaheuristic approaches at the expense of higher computational time.

As for the application of strict single metaheuristic methodologies, PSO was found to be competitive with all three of the previously employed algorithms (GA, TS, and SA) for the cardinality-constrained portfolio selection problem and especially successful in low-risk portfolios (Cura, 2009). To evaluate the performance of PSO for even more realistic instances, Golmakani and Fazel (2011) further introduced minimum transaction lots, bounds on holdings, and sector capitalization in addition to cardinality constraints. These authors applied a combination of binary PSO and improved PSO (CBIPSO), and found that CBIPSO outperforms GA in that it provides better solutions in less computing time, especially for large-scale problems. As constraints become increasingly complex, the question of constraint-handling in determining feasible solutions arises. Reid and Malan (2015) investigated this research line and developed a portfolio repair constraint handling technique applied in a PSO portfolio optimization. Employing this, they were able to further improve the performance of the metaheuristic, again particularly for large instances.

Di Tollo and Roli (2008) provided a survey concerned with the early applications of metaheuristics to the POP and some of the proposed constraints explicitly highlighting the potential use of hybrid approaches. Likewise, such a hybrid method was proposed by Maringer and Kellerer (2003), who employed a hybrid local search algorithm combining principles of SA and EA to optimize a cardinality-constrained portfolio. By combining exact mathematical programming and metaheuristic methods, Woodside-Oriakhi et al. (2011) further hybridized and created different matheuristics. This option was also investigated by Schaerf (2002) and Di Gaspero et al. (2011) who respectively combined TS and first descent (FD) and steepest descent (SD) local search metaheuristics with quadratic programming to optimise a portfolio while accounting for cardinality

constraints, lower and upper boundaries for the quantity of an included asset, and pre-assignment constraints. According to their results, the developed solver finds the optimal solution in several instances and is at least comparable to other state-of-the-art methods for the others. Concerning optimality, Cesarone et al. (2013) were able to develop an exact increasing set algorithm that, for small instances, solves the POP with quantity and cardinality constraints optimally and can be extended into a heuristic procedure to account for larger instances. It outperforms the metaheuristics employed by Di Gaspero et al. (2011) and Schaerf (2002) in all instances.

Multi-objective portfolio optimization

Multi-objective optimization methods combine two objective measures into a single one that is to be optimized (Mishra et al., 2014) or, more often, find a set of Pareto solutions while balancing two or more objective functions simultaneously. With respect to single-objective optimization methods that require the *ex-ante* definition of an acceptable degree of profitability, multi-objective optimization requires no previous knowledge about the investor's degree of risk aversion and is thus a more general approach transferrable to different decision-makers. The approach of combining risk and return characteristics into a single objective function is taken by Zhu et al. (2011). They introduced the Sharpe ratio as a simultaneous measure and, since GA and PSO have been found to be competitively successful in solving the single-objective version, performed a comparison of these metaheuristics in solving the non-linear constrained portfolio optimization problem. As previously established, they also argue that PSO outperforms GAs, especially in large instances. While they did not include realistic constraints other than a total portfolio weight equal to one in addition to portfolio assets restricted to positive weights, in which the short selling of the portfolio's underlying assets is prohibited, the authors also investigated unrestricted portfolios. The solution portfolios obtained with the PSO solver outperformed those constructed using GA for all test problems in terms of Sharpe ratio, and the established efficient frontier was above that of GA portfolios in all but one instance.

According to Streichert et al. (2003), the multi-objective POP can be formulated employing two simultaneous objective functions as follows. For a multi-objective optimization it becomes necessary to minimize the portfolio risk expressed by the portfolio variance:

$$Min \sum_{i=1}^{N} \sum_{j=1}^{N} w_i w_j \sigma_{ij}, \tag{8.4}$$

while maximizing the return of the portfolio, i.e.:

$$Max \sum_{i=1}^{N} w_i \mu_i, \tag{8.5}$$

subject to:

$$\sum_{i=1}^{N} w_i = 1, \tag{8.6}$$

$$0 \leq w_i \leq 1, \quad \forall i \in \{1, 2, \ldots, N\}. \tag{8.7}$$

Alternatively, Equations 8.4 and 8.5 can be combined into a single one by incorporating objective weights as follows (Mishra et al., 2014):

$$Min\ \lambda \sum_{i=1}^{N} \sum_{j=1}^{N} w_i w_j \sigma_{ij} - (1 - \lambda) \sum_{i=1}^{N} w_i \mu_i, \tag{8.8}$$

subject to the aforementioned constraints. In this case, the weights as determined by the parameter λ represent the risk aversion of the investor. By varying this parameter and running repeatedly, a Pareto efficient frontier can be established. Because of the high performance of PSO in solving the single-objective POP, enhanced PSO algorithms for solving the multi-objective POP have been proposed by Deng et al. (2012) and He and Huang (2012). Cardinality and bounding constraints were incorporated by Deng et al. (2012) who find that their algorithm mostly outperforms GA, SA, and TS algorithms as well as previous PSO approaches, especially in the case of low-risk portfolios. It can be concluded that different findings unanimously favour PSO in situations when low-risk investment is demanded in addition to a larger-scale potential asset pool. Similarly, He

and Huang (2012) proposed a modified PSO (MPSO) algorithm that outperforms regular PSO for their four optimization sets. More recently, they also developed a new PSO to deal with discontinuous modelling of the POP and find that it generally outperforms PSO and also performs better than MPSO in larger search spaces (He and Huang, 2014). Other population-based algorithms applied in optimizing cardinality-constrained portfolios include firefly algorithms (FA) (Tuba and Bacanin, 2014b) and artificial bee colony (ABC) algorithms (Tuba and Bacanin, 2014a). However, because the results were satisfactory at most even after modifications, the authors hybridized FA and ABC by incorporating the FA search strategy into ABC to enhance exploitation and found that their data suggested the superiority of the methodology compared to GA, SA, TS, and PSO (Tuba and Bacanin, 2014a). Streichert et al. (2003) accounted for further constraints: buy-in thresholds (acquisition prices) and round lots (smallest volume of an asset that can be purchased). They employed two MOEAs: a GA and an EA enhanced through the integration of a local search that applies Lamarckism, thus allowing the individual improvements to be passed on to the offspring. They found that this enhancement greatly improved the reliability of the results, especially with respect to the additional constraints. Unfortunately, these approaches are hardly reproducible due to their complexity, reinforcing the need for a less metaphorical and more scientifically reproducible approach.

Nevertheless, apart from the neglect of realistic non-linear constraints, there is a second point of criticism to the original Markowitz model, namely its assumption of normal financial returns, which, in reality are characterised by a leptokurtic distribution (Krink and Paterlini, 2011), making it necessary to consider non-parametric risk measures. Such a measure is the value-at-risk, as employed by Babaei et al. (2015) who developed two multi-objective algorithms based on PSO to solve a cardinality- and quantity-constrained POP. Through splitting the whole swarm into sub-swarms that are then evolved distinctly, their methodology outperformed similar benchmark metaheuristics. In order to optimize a non-parametric value-at-risk and to include further constraints, including lower and upper bounds for the weights of included assets, a threshold for asset weight changes, lower and upper bounds for the weights of one asset class and a turnover rate that determines the maximum asset allocation changes possible at once, Krink and Paterlini (2011) developed the differential evolution (DE) for multi-objective portfolio optimization algorithm. An extended version of a generalised DE metaheuristic was also employed in optimizing a highly constrained POP by Ayodele and Charles (2015). The included constraints consist of bounds on holdings, cardinality, minimum transaction lots, and expert opinion. An expert can form an opinion based on indicators beyond the scope of the analysed data and influence whether or not an asset should be included. Their methodology showed improved performance when compared to GA, TS, SA, and PSO. Lwin et al. (2014) considered cardinality, quantity, pre-assignment and round lot constraints and developed a MOEA that is improved through a learning-guided solution generation strategy, which promotes efficient convergence . It was shown that the developed algorithm outperformed four benchmark state-of-the-art MOEAs in that its efficient frontier was superior.

An extensive review of the application of EAs to the POP is provided by Metaxiotis and Liagkouras (2012). Likewise, for an extensive review on different POPs, including single- and multi-objective optimization, the reader is referred to Mansini et al. (2014). It can be asserted that population-based metaheuristics have yielded superior results compared to single-solution metaheuristics in the case of single-objective portfolio optimization. This has resulted in them being dominantly applied to multi-objective portfolio optimization.

Passive investment

Closely related to portfolio optimization as an active portfolio management strategy, passive investment strategies have received less attention in the optimization literature. These strategies are characterized by limited on-going buying and selling, as well as by ensuing limited maintenance. Based on the traditional capital market theory stating that market portfolios offer the greatest return per unit of risk, passive investment strategies have been shown to outperform actively managed funds and thus gained popularity (Alexander and Dimitriu, 2004).

Index tracking

The index tracking problem (ITP) is a passive portfolio management strategy in that investors aim at mimicking a market or sector index. This is done by either replicating the index or by selecting a portfolio that follows the index behaviour as closely as possible without including all the stocks that make up the original index. In the case of perfect replication, there are transaction

costs associated with updating the portfolio to continuously accurately depict the index. Therefore, the ITP is largely concerned with the latter, partial replication. There are thus two stages in index tracking, the common goal of which is to minimize the resulting tracking error (the distance between the portfolio and benchmark returns). The first consists of selecting the assets to include in the portfolio and the second relates to determining the weights. Thus, it consists of a combinatorial and a continuous numerical problem, which both have to be addressed simultaneously (Krink et al., 2009). Once similar constraints as in portfolio optimization are introduced (e.g. floor and ceiling constraints, cardinality constraints, pre-assignments, or class constraints), minimizing the objective function of the tracking error becomes extraordinarily difficult to solve with exact methods.

The optimization problem can thus be addressed with the following formulation (Beasley et al., 2003). Minimize the tracking error:

$$Min\ E = \frac{\left[\sum_{t \in S} |r_t - R_t|^{\alpha}\right]^{\left(\frac{1}{\alpha}\right)}}{T}, \tag{8.9}$$

where $S = 1, 2, \ldots, T$ are the time periods considered during which the portfolio return was below that of the tracked index, r_t is the tracking portfolio return, R_t is the return of the tracked index itself, and α is the penalization power that is applied to the difference between the realized return and the benchmark return. If we set $\alpha = 2$, the tracking error is defined as the root mean square error (RMSE). In the case of a perfect reproduction of an index, the tracking error would naturally be equal to zero. In the most basic formulation, the following constraints have to be considered:

$$\sum_{i=1}^{N} z_i = K, \tag{8.10}$$

which represents the cardinality constraint and ensures that any new tracking portfolio contains K stocks, as z_i takes on the value of one if a stock is included in the replication portfolio and zero otherwise. The weights have to be limited:

$$0 < w_i \leq 1,\ z_i = 1,\ \ \forall i \in \{1, 2, \ldots, N\}, \tag{8.11}$$

This limits the weights of the included stocks to be larger than zero and equal to or below one. The non-included stocks must naturally dispose of a weighting of zero:

$$w_i = 0,\ z_i = 0,\ \ \forall i \in \{1, 2, \ldots, N\}. \tag{8.12}$$

Maringer and Oyewumi (2007) investigated partial replication and introduced cardinality constraints concerning upper and lower weight limits and integer constraints in the ITP employing a DE methodology. Their findings suggest that partial replication is indeed sufficient in replicating the benchmark index. This is due to the fact that only a decreasing marginal improvement is reached by increasing the cardinality.

Scozzari et al. (2013) were able to develop a mixed integer quadratic programming formulation to solve the ITP including hard constraints set by the European Union on ceilings of asset inclusion weights as well as low turnover rates and resulting low transaction costs in small instances. However, the introduction of realistic constraints generally makes it difficult to use exact methods in solving large ITP instances. Early research by Beasley et al. (2003) introduced a population-based evolutionary metaheuristics to solve the partial reproduction ITP with regard to stock indices including constraints on transaction costs (as well as a ceiling for the total inclusion of stocks). Derigs and Nickel (2004) developed a two-stage SA metaheuristic, in which they controlled for cardinality constraints and transaction costs through turnover volume restrictions.

For larger instances, especially in multi-period analysis, Scozzari et al. (2013) proposed hybridizing metaheuristics with exact methods. This has been done by Krink et al. (2009) who addressed the two subtasks of ITP simultaneously and applied a metaheuristic approach based on DE combined with a combinatorial search operator. Although their developed methodology initially failed to find acceptable solutions, they showed that extending DE with a search operator by selecting the assets with highest weights in the benchmark improved the results greatly in comparison with GA, SA, and PSO. Ruiz-Torrubiano and Suárez (2009) employed a GA hybridized with quadratic programming. More recently, Ni and Wang (2013) also tackled the ITP employing

a hybridized GA with increased learning ability that is enabled through goal programming. The authors included cardinality and integer constraints, as well as proportion constraints for individual portfolio assets. While both methodologies yielded successful solutions, the models neglect transaction costs. The trade-off between transaction costs and tracking performance was then investigated by Chiam et al. (2013) who developed a multi-objective evolutionary index tracking platform that considers multiple periods and simultaneously optimizes tracking performance and transaction costs while considering round lots and non-negativity constraints as well as floor constraints as buy-in threshold to prevent unnecessary transaction costs and capital injections.

Although different metaheuristic approaches have been chosen to cope with the realistic constraints of the ITP, Affolter et al. (2016) found that due to the missing measure to define the distance between portfolios with respect to their assets and weights, invasive weed optimization (IWO) did not lead to satisfactory optimization results. Di Tollo and Maringer (2009) created a framework for classifying the metaheuristics applied to ITP and present a review of the literature.

Enhanced index tracking

Beasley et al. (2003) defined an objective function that accounts for a trade-off between the tracking error and excess returns above those of the benchmark index. This enhanced index tracking allows the manager discretion in pursuing risk-limited active strategies to enhance return. Considering that investors might see a trade-off between the trading error and excess returns above the index has led to the enhanced index tracking problem (EITP), in which investors aim at beating the benchmark index. This can either be done through active selection of the included assets and weights or through a passive extension of the methodology by incorporating the excess return as a further optimization objective. The EITP then becomes a multi-objective optimization problem, in which the tracking error is minimized while maximizing the degree of beating the benchmark index so that a solution dominates another if the excess return is higher given the same level of trading inaccuracy or if the trading accuracy for the same level of excess return exceeds that of the other solution. This can be formulated by including a second objective function that defines the excess return between r_t and R_t:

$$Min\ E = \frac{\left[\sum_{t \in S} |r_t - R_t|^{\alpha}\right]^{\left(\frac{1}{\alpha}\right)}}{T}, \tag{8.13}$$

while maximising the excess return r^*:

$$Max\ r^* = \sum_{t=1}^{T} \frac{r_t - R_t}{T}, \tag{8.14}$$

subject to the aforementioned constraints:

$$\sum_{i=1}^{N} z_i = K, \tag{8.15}$$

$$0 \leq w_i \leq 1, z_i = 1, \quad \forall i \in \{1, 2, \ldots, N\}, \tag{8.16}$$

$$w_i = 0, z_i = 0, \quad \forall i \in \{1, 2, \ldots, N\}. \tag{8.17}$$

Canakgoz and Beasley (2009) solved the ITP as well as the EITP including transaction costs, an upper limit on the total number of stocks purchased, and a limit on the incurred transaction costs using exact methods (mixed-integer linear programming formulations). However, Li et al. (2011a) showed they could mostly outperform the methodology employed by Canakgoz and Beasley (2009) by implementing an immunity-based optimization algorithm. It is an EA based on the clonal selection of an immune system, or the immune response to antigens (De Castro and Von Zuben, 2002). Including further constraints, Li and Bao (2014) also employed an immunity-based multi-objective optimization algorithm with non-negativity and floor and ceiling buy-in thresholds. They concluded that the inclusion of optimization of the tracking process in addition to optimizing tracking error and excess return is valuable as the optimization of the tracking process improves results in most instances. A perfectly enhanced tracking portfolio would outperform the index by a low-frequency trend such as steady excess return while negative returns should be trendless and characterised by high frequency variation. Thus, the tracking process can be enhanced by considering different frequencies for tracking error and excess returns when the former is minimized

and the latter maximized (Li and Bao, 2014). Optimization of the tracking process is expected to increase in importance for multi-period assessment; the authors, however, leave this for further research. The question of multi-periodicity was investigated by Andriosopoulos et al. (2013) who addressed the EITP employing both DE and GA. They could show that the so-constructed mimicking portfolios inhibit less risk compared to the underlying benchmark index, while proficiently replicating their performance. Nevertheless, they concluded that the GA version outperforms DE in terms of minimum tracking errors, as well as maximum mean excess returns. As they explicitly considered different time horizons for rebalancing the portfolio, these authors reinforced the idea that there exists a trade-off between transaction costs, which decrease with longer rebalancing periods, and Sharpe ratios (as a measure of the tracking performance and profitability), which is negatively impacted by decreased rebalancing frequency as investigated by Chiam et al. (2013) for the ITP.

An alternative approach was pursued by Guastaroba and Speranza (2012) who applied a kernel search framework to both the ITP and the EITP. They argued that error measurements should be undertaken as absolute values and introduced the possibility that an investor already holds a portfolio as a further constraint to consider in addition to transaction costs. However, they treated the EITP as a single-objective optimization by outperforming the market index, while keeping the tracking error below a given threshold. Compared to a general-purpose solver, the performance of the kernel search model was superior. Thomaidis (2011) considered an EITP problem with restrictions on the maximum of tradable assets, and employed fuzzy set theory to consider non-standard investment objectives, such as the probability of under-performing. The resulting cardinality-constrained problem was solved using nature-inspired optimization techniques: SA, GA, and PSO.

Lastly, while some authors declare active and passive portfolio management as mutually exclusive concepts, the close connection between index tracking and portfolio optimization could be illustrated by the approach taken by Di Tollo et al. (2014) who combined the two methods in a multi-criteria optimization problem. They employed a hybrid metaheuristic consisting of local search metaheuristics (FD, SD and TS) and quadratic programming to estimate the efficient frontier. Combining the concepts of risk and return with tracking error led to a three-dimensional objective function and Pareto frontiers. The developed methodology was found competitive in performance with other metaheuristics such as TS.

Project portfolio selection

Unlike banks and institutional investors, non-financial companies as well as governments are faced with a different type of portfolio choice. As a method to determine which proposals to pursue and the corresponding budget allocation, investment or project appraisal is related to portfolio optimization in its goal of maximizing a benefit figure. Usually, decisions cannot be altered or adjusted during the course of the projects, or at least not without incurring considerable financial losses. Thus, investment appraisal determines a strategic organizational path for the medium and long term. This problem becomes $\mathcal{NP}$-hard due to its sheer complexity (Fernandez et al., 2015). It is by its very nature a multi-period problem and the budget-allocating entity usually pursues several conflicting objectives, some of which can be of qualitative nature. For that matter, Doerner et al. (2004) proposed a two-stage procedure. During the first phase, the Pareto frontier is constructed. Then, in the second phase, it is interactively explored by the decision-makers to account for personal preferences. A formal description of this problem, based on the one presented in Doerner et al. (2004), is included next. The benefit function $b_{lt}(x)$ that comprises the value of the l different benefit groups, such as generated funds, cash flows, patents or other beneficial outcomes of the selected projects is to be maximized over all considered time periods t for all included projects, i.e.:

$$b_{lt}(x) = \sum_{i=1}^{N} b_{ilt} x_i, \tag{8.18}$$

where x_i is a binary variable that takes on the value of one for included projects and zero otherwise, subject to constraints concerning resource limitations R_{qt} that apply to all resource categories r_q, such as budget, capacity, or manpower, as well as minimum benefit requirements B_{lt}

that define a threshold below which the decision-maker is uninterested in the implementation of projects:

$$r_{qt}(x) \leq R_{qt}, \forall q \in \{1, \ldots, R\} \text{ and } \forall t \in \{1, \ldots, T\}, \tag{8.19}$$

$$b_{lt}(x) \leq B_{lt}, \forall l \in \{1, \ldots, B\} \text{ and } \forall t \in \{1, \ldots, T\}. \tag{8.20}$$

Because of the modelled similarities, the methodological approaches employed are inspired by the research on traditional portfolio optimization. Early work (Ghasemzadeh and Archer, 2000) conducted optimization after the construction of a weighted objective function and constraints concerning budget and man-hours in an integer linear programming approach. However, test instances were very limited because the authors aspired a comparison between manually computed portfolios and those constructed employing their decision support system. For their metaheuristic two-stage approach Doerner et al. (2004) employed Pareto ACO (P-ACO). As there are possible synergies between projects that should be evaluated in order to accurately estimate the benefits of a project portfolio, the authors made an attempt at incorporating these considerations into their methodology and pointed out that, unlike GA, SA, and TS that are adaptive metaheuristics, P-ACO specifically constructs project portfolios through pheromone vectors. This has two advantages. Firstly, infeasible solutions are avoided and secondly, project interactions can more naturally be considered in the construction of solutions. They further took into account floor and ceiling constraints for inclusion of projects from any given subset, as well as resource limitations and minimum benefit requirements for individual projects. Compared to Pareto SA and a non-dominated sorting GA (NSGA), P-ACO yielded the most efficient results. This approach was then further enhanced by Stummer and Sun (2005), who compared the performance of a P-ACO procedure enhanced through adding a neighbourhood search routine, a TS procedure, and a variable neighbourhood procedure. Their findings suggested that the improved P-ACO model performs better than TS with many objective functions and a large set of efficient solutions and is thus specifically suitable for real-life problems. Furthermore, Doerner et al. (2006) concluded that including both a learning and a two step integer linear pre-processing procedure to initialize several initial efficient project portfolios improves performance of the P-ACO algorithm.

More recently, research has also drawn on findings from other areas, such as scheduling: Gutjahr et al. (2008) and Gutjahr et al. (2010) also took employee competencies and the evolution of their knowledge scores over time through learning or depreciation into account. While the earlier work optimized a weighted average objective function using ACO and GA metaheuristic procedures and found the GA to be superior when the search space is not highly constrained, the authors developed a multi-objective optimization model, which simultaneously optimizes the objectives of maximum economic gains and aggregated competence increase in their later work. They also divided the problem into master and slave subproblems, the first of which is concerned with the project selection, while the slave problem optimizes the allocation of personnel to the projects over time. Although the slave problem can be solved using exact methods, the master problem was solved using the NSGA-II and P-ACO metaheuristics. While both performed reasonably well, NSGA-II outperformed P-ACO in synthetic test instances, while P-ACO outperformed NSGA-II for the investigated real-life instances. Carazo et al. (2010) further investigated this research line and included scheduling as a continuative concept following the project selection. Their developed metaheuristics approach is based on SS for project portfolio selection (SS-PPS). As previous work, they also considered interdependences between different projects and can show that their model outperformed other heuristic approaches based on EA (SPEA). Similar to Rabbani et al. (2010), who presented a multi-objective PSO metaheuristic and found it to be competitive with respect to SPEA II, Urli and Terrien (2010) formulated the project portfolio selection problem as a multi-objective non-linear integer program, which they solved using the SSPMO metaheuristic (Molina et al., 2007). In a first phase, they generated an initial set of efficient solutions through TS and then combined these via SS. While this approach solved small and medium instances in satisfactory computation time, the determination of all non-dominated project portfolios still remains difficult when considering large, but realistically relevant instances (100 projects or more).

Another issue that has only recently been addressed is project divisibility. While business projects are at least partially indivisible, research projects funded by governments can often also be executed with partial funding and it is thus a further question how much of the sought after funding is awarded, introducing further constraints to the budget allocation. Cruz et al. (2014)

used ACO in solving a stationary project portfolio optimization problem, in which partial support of the requested budget was allowed. They developed a non-outranked ACO approach, incorporating a fuzzy outranking preference model, and assumed that the preferences of the decision-maker are to some extent known. Outranking was employed in an *a priori* preference system in order to model that decision-makers will have preferences towards different portfolios on the efficient frontier based on their personal goals concerning the achievement of objectives. Fernandez et al. (2015) further enhanced this approach by including integer linear programming methods to generate an initial population. They also included synergies in their optimization, concluding that their model outperformed state-of-the-art metaheuristics. It can be asserted that project synergies, project divisibility, the incorporation of multi-periodicity, and outranking are the prominent real-life constraints and trends that specifically increase the complexity of the portfolio selection process.

8.2.2 Risk management

Risk management of companies refers to the evaluation of realistic data concerning the institution's exposure to a certain source of risk and it is further concerned with statistics on trends that will influence that exposure in the future. While quantitative data is relevant and necessary for this, it must be complemented by qualitative information for informed decision-making (Chorafas, 2007). Risk management is addressed in terms of optimization through metaheuristics for credit risk assessment and the resulting bankruptcy prediction. García et al. (2015) provide a review of systems and applications to the optimization of trading rules in the financial markets. Table 8.2 presents the metaheuristic methodologies applied to the different subproblems of risk management.

Table 8.2: Application of metaheuristics and hybridization to risk management.

Optimization problem	Single-solution search		Population-based search										Hybrid
	SA	TS	GA	ACO	EA	ABC	PSO	SS	HBMO	FA	BA	HS	
Credit risk assessment		2	4	1			1	1	1				6
Bankruptcy prediction			4		1		2						6
Optimization in stock trading	2		4			1	2			1	1	1	7
Optimization in foreign exchange trading			3		1								4

From Table 8.2, several conclusions can be drawn. Firstly, GA are the preferred metaheuristics in risk management as well. Furthermore, PSO has also received widespread attention. Contrary to that, more exotic algorithms, such as harmony search (HS), FA, or bat algorithms (BA). Secondly, it can be seen that bankruptcy prediction, as well as optimization of trading systems for foreign exchange markets, have received less attention in the literature and have been approached with fewer methodologies. They thus represent interesting future research lines. Thirdly, it becomes evident that hybridization among metaheuristics or other optimization methods is far more prevailing in risk management optimization than in portfolio optimization. Lastly, it is evident that relatively recently developed metaheuristics, such as IWO and honey bees mating optimization (HBMO), have not been applied as comprehensively as well-established ones.

Credit risk assessment and optimization

Credit risk assessment is one of the most researched and recognized topics in the banking industry. There are many different approaches for financial institutions. However, during the last years, non-financial companies have also recognized the need to treat their trade credits to customers with the same caution and scrutiny. While the use of metaheuristics is still scarce, they are increasingly used as a pre-processing procedure in order to identify the most relevant predictors of credit risk in the analysis of large datasets. Marinakis et al. (2008) classified a set of companies into different classes of credit risk level. They propose and compare TS, GA, and ACO for solving the feature selection subset problem, which are then used in determining the appropriate level of credit risk. The employed accuracy measures are determined by whether or not a subject has been classified in the right category (Table 8.3).

TABLE 8.3: Definitions of the classified and the misclassified samples.

		Actual class	
		1	2
Estimated class	1	T_1	F_2
	2	F_1	T_2

The overall classification accuracy (OCA) can then serve as optimization objective that is to be maximized:

$$Max\ OCA = \frac{T_1 + T_2}{T_1 + F_1 + T_2 + F_2} * 100. \tag{8.21}$$

More recently, Marinaki et al. (2010) employed HBMO in determining the relevant features. They were able to show that this metaheuristic reduced the number of used features by more than half and still yielded superior results compared to PSO, ACO, GA, and TS. Oreski et al. (2012) employed NN hybridized with GA (GA-NN) to enhance the classification accuracy of the NN classifiers by choosing optimal features. Oreski and Oreski (2014) further improved the results by employing a hybrid GA instead of GA. Their results suggested that they hence achieved a higher and less volatile accuracy with on average fewer selected features through a reduction of the search space and an incremental phase of the GA. Chi and Hsu (2012) employed GA in selecting relevant variables to combine a bank's internal behavioural scoring model with an external credit bureau scoring model and thus creating a dual scoring model that outperformed the individual model. A survey on the importance of employing the right fitness function in the GA for credit assessment is provided in Kozeny (2015).

Trends in credit risk assessment concern the hybridization of metaheuristics with other techniques for feature selection. Wang et al. (2010) developed a feature selection based on rough set and TS. In comparison with non-preselecting models, the savings in computational time and performance accuracy were significant. Similarly, Wang et al. (2012) used a rough set and scatter search feature selection that is able to improve results in all three considered base sets, i.e. NN, J48 decision tree and logistic regression (LR). Lastly, Danenas and Garsva (2015) pursued the idea of optimizing the classifiers of a linear SVM using PSO. While their results were comparable to the use of other classifiers (LR and radial basis function or RBS networks), the proposed methodology, however, delivered less stable performance.

Bankruptcy prediction

Bankruptcy occurs when debtors are unable to repay outstanding debts. While bankruptcy prediction constitutes part of the credit risk evaluation process, it is vital for banks and companies to constantly monitor their credit risk exposure. Because of the two-classes framework (firms that go bankrupt and firms that do not), the basic optimization framework is similar as suggested for credit risk assessment. The difficulty and difference lies in the relatively longer aspired forecasting period and the difficulty in predicting the exact time of bankruptcy.

It is worth considering to differently value the two classes of mistakes that occur. While falsely classifying a subject as bankruptcy candidate (type II misclassification) merely leads to missed revenues, a false classification as healthy company (type I misclassification) usually leads to at least partial failure on a payment and thus has greater consequences for profitability.

Early research conducted by Back et al. (1996) highlighted the contribution of GA in predicting bankruptcy when hybridized with NN. Shin and Lee (2002) introduced the prediction of corporate bankruptcy using GA and historical financial data. Kim and Han (2003) further employed GA to extract decision rules based on qualitative expert decisions and find their methodology to be superior compared to NNs or inductive learning methods because the rules created by GA are more accurate and have larger coverage. An extensive survey on the early research in this knowledge area can be found in Kumar and Ravi (2007) who reviewed both statistical and computing methods. Their evaluation concluded that all statistical methods are outperformed by back propagation NNs. They further highlighted the prediction accuracy of SVM. More recently, Kirkos (2015) presented the literature on artificial intelligence and machine learning techniques employed in bankruptcy prediction.

Min et al. (2006) improved the performance of SVM with regards to optimizing the feature subset and parameters simultaneously. They showed that selecting an appropriate feature subject has implications for the kernel and that their integration improved bankruptcy prediction accuracy. Chen (2011) highlighted that while intelligent techniques provide higher prediction accuracy for smaller datasets and are adversely affected by increasing datasets, statistical methods perform more accurately when the dataset is large. But the author also indicated that a hybrid between PSO and SVM could yield a good balance between short- and long-term prediction accuracy. This was consequently done by Lu et al. (2015) who combined switching PSO (SPSO) and SVM. The SPSO was employed in searching the optimal parameter values of RBF kernel of the SVM and the authors showed that this hybridization yielded superior results to GA-SVM and PSO, respectively. These findings were supported by Chen (2011) and Chen (2014a) who also employed PSO-SVM and showed high accuracy with a significantly reduced number of parameters. Furthermore, Gaspar-Cunha et al. (2014) proposed an evolutionary multi-objective approach that simultaneously minimizes the number of features and maximizes the accuracy of the classifier in SVM so that the algorithm is self-adaptive. The general advantage of multi-objective optimization lies in the attainment of a set of efficient solutions from which the decision-maker can perform a trade-off based on personal preferences.

Recently, ensemble learning has been applied to the bankruptcy prediction problem. Kim and Kang (2010) proposed hybridizing an ensemble with NNs and showed that it improved prediction accuracy compared to regular NNs. However, these attempts often suffer from high correlation among the individual classifiers, and thus Kim and Kang (2012) improved their methodology to include a GA-based coverage optimization to alleviate multicollinearity through classifier selection. More recently, Davalos et al. (2014) developed an accurate GA-based ensemble classifier model with heterogeneous instead of individual classifiers that is comprehensible due to its if-then-structure. They showed the improved performance of their approach.

However, the financial ratios employed in the main research lines are unavailable for a large portion of companies. Small and medium-sized enterprises (SMEs) do not dispose of regular audited financial data or market prices and public ratings due to publicly traded equity or debt instruments and it is necessary to include available and relevant indicators for these individual firms. Thus, with special regards to SMEs, Gordini (2014) compared the prediction accuracy of GA, SVM, and LR. The author showed that the prediction of GA was superior, especially with regard to type II misclassifications and with regard to prediction of bankruptcy for small firms.

Optimization of decision-support systems for trading

The development and optimization of automated trading systems has become of prevalent importance and special interest for broker investment banks and other institutional investors alike. A large portion of the literature addresses stock trading, while some researchers have concentrated on the foreign exchange markets.

Stock market trading Derigs and Nickel (2003) developed a decision support system (DSS) for portfolio optimization and index tracking. They stressed the importance of hard (government-imposed and compulsory) and soft (shaped by preferences of the investor) constraints. They implemented a local search guided by SA in order to optimize the DSS with respect to floor and ceiling constraints and transaction costs. These authors have shown for the application to passive tracking of the DAX 30 that their system delivered solutions with minimal tracking errors in acceptable computing time. Focusing on real-time decisions, Chavarnakul and Enke (2009) proposed a trading system for the stock market based on volume adjusted moving average that is hybridized with NN to decrease the time of receiving trading signals, fuzzy logic to deal with uncertainty, and GA techniques to optimize the trading signals to overall increase efficiency. Depending on the strength and direction of a given signal, the system assumes a buy or sell position. If the signal is not confident enough, a hold position is taken. The so established neuro-fuzzy based GA was shown to lead to fewer trades and thus reduced transaction costs, while profitability was increased.

Gorgulho et al. (2011) also proposed a system to automatically manage a portfolio of assets and highlighted the necessity of adapting the system to the state of the market. They employed GA and technical analysis rules. The system requires the user to input the available budget, the maximum of assets to be included in the portfolio at any time, whether or not short selling is considered and the allowed amount of transaction costs. Then, an initial portfolio is constructed employing a GA. Over the course of the investment, the system regularly updates the proposal based

on technical trading rules based on closing positions and refilling empty positions. Teixeira and De Oliveira (2010) combined technical trading rules with nearest neighbour classification. Their analysis was based on historical data of stock closing prices and volume, on the basis of which trading rules were formed. Their system outperforms a buy and hold strategy in most cases. Because the parameters in these functions have to be determined, metaheuristics have been applied. The hybridization of technical trading rules and metaheuristics is seen as an especially promising research area. Brasileiro et al. (2013) further refined the strategy by Teixeira and De Oliveira (2010) by searching for the best system parameters and number of lags with an ABC algorithm and thus outperforming the previous trading system as well as the buy and hold strategy in most instances. Nunez-Letamendia (2007) had already shown that GA is robust and powerful when applied to optimizing technical trading rules. Similarly, Lin et al. (2011) applied a GA to improve trading rules for individual stocks, which are then combined with echo state networks to provide suggestions for trading. Their results showed an outperformance of the buy and hold strategy. In a more recent work, Wang et al. (2014a) employed a time-variant PSO (TVPSO) to determine the optimal parameter values of a complex trading system: performance-based reward strategy (PRS). PRS combines moving average and trading range breakout trading rules. Considering transaction costs and excluding short selling, the system was able to achieve high profits and the application of the TVPSO significantly optimized the trading system.

Hybridizations of metaheuristics and NN have recently shown to provide accurate forecasts of stock market prices. While both provide better results than a passive buy and hold strategy, HS based models have been shown to outperform GA based models with regards to forecasting errors (Göçken et al., 2016). Very recently, the hybridization of data mining techniques with metaheuristics has created clustering metaheuristics able to predict patterns in the general movement of stock markets, such as periods of lower and higher return (Prasanna and Ezhilmaran, 2015).

Foreign exchange market trading

Foreign exchange market trading can either concern hedging foreign exchange risk or speculation. Only the latter has the objective of making a profit by exploiting market inefficiencies. Myszkowski and Bicz (2010) established a trading system based on decision trees that consider technical trading indicators. EA then generates trading strategies. While these were able to achieve a profit, the system is still too fragile to be used in automated trading. Mendes et al. (2012) proposed employing GA to optimizing trading rules and although their developed trading system performs well in terms of computational time because of the small population size employed, it fails to perform well in terms of profitability when faced with transaction costs. Zhang and Ren (2010) developed a high-frequency trading system based on the optimization of technical indicators through GA that was able to produce annualised profits. In addition to intraday prediction, Evans et al. (2013) implemented a trading system based on NNs, whose topology was optimized using GA. In comparison with Zhang and Ren (2010), they were able to considerably improve annualized net profits.

8.2.3 Linkage between portfolio optimization and risk management

Despite the fact they have been addressed as two independent types of problems in most of the scientific literature, this section highlights the relationship between portfolio selection and risk management. In the first place, POPs directly consider a risk measure (such as portfolio variance, portfolio semivariance, value at risk, alpha and beta, among others) and, therefore, they can also be seen as risk management problems. For example, the specification of adequate risk measures that accurately depict return distributions is a well-established area of research of on-going interest. It is concerned with one-dimensional risk measures of individual assets and multi-dimensional measure to account for interaction of asset portfolios (Rachev et al., 2010).

In the second place, most risk management models related to optimization problems can be seen as rich variants of POPs. For instance, stock market trading is in the essence of the problems concerned with constructing an initial portfolio and updating it over time to reflect current macro- and microeconomic developments. Likewise, while credit risk and bankruptcy risk are only estimated in the overwhelming majority of the risk management literature, it is the ultimate goal to build low-risk portfolios by including preferably those assets with a lower credit risk and excluding other assets expected to go bankrupt. Using a MOEA, Moreno-Paredes et al. (2013) explicitly acknowledge the linkage between credit risk management and portfolio optimization by treating the loan decision among a set of customers as a credit portfolio optimization problem.

More generally, implicit in a POP is pooling assets with imperfectly correlated returns that leads to a diversification of idiosyncratic sources of risk and a reduction in the overall risk of portfolio investment.

Figure 8.3 depicts the relationship between risk management and POPs reviewed in this section. The extension of the ovals representing risk management problems and portfolio optimization problems respectively signifies the extension of possible solving approaches beyond traditional optimization techniques. The risk management problems of bankruptcy and credit risk prediction are located directly on the border to portfolio optimization, as the predictions are generally employed in a following portfolio selection process. Foreign exchange trading, unlike stock trading, is considered a sole risk management problem. While stock trading consists of the establishment and maintaining of a portfolio, speculative profits in foreign exchange trading are generated through simultaneous buying and selling and not the establishment of a portfolio. Concerning the subproblems of portfolio optimization, the ITP and EITP do not directly consider risk measures. Unlike that, the POP is directly concerned with risk minimization and thus closely related to risk management problems.

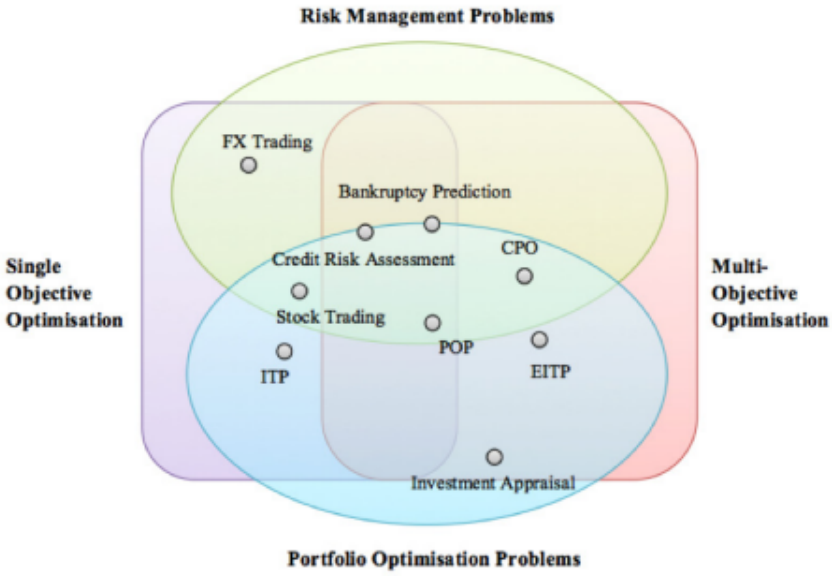


FIGURE 8.3: A unified classification of portfolio optimization and risk management.

8.2.4 Emerging trends

From the previous discussion, one clear trend is the transfer of methodological knowledge from portfolio optimization to risk management optimization. Another trend is related to the increasing complexity of the problems being addressed, which makes even more evident the need for faster (or parallelizable) metaheuristic approaches. These 'fast metaheuristics' will be needed as the models introduce further constraints to account for real-life circumstances, and as the real-time factor that is required in most of the decision-making processes will become even more relevant. Strongly related to this point, distributed and parallel computing techniques can be employed to accelerate the 'wall clock times' necessary to obtain near-optimal solutions to large-scale problems (Juan et al., 2013b). Furthermore, the fact that two or more objectives have to be considered simultaneously to account for the complexity has shown to increase the employment of multi-objective optimization techniques.

Another clear trend is the predominance in the use of population-based metaheuristics over single-point metaheuristics. It is our opinion that no family of metaheuristics are shown to be superior in performance (regarding both quality of solutions as well as computing times) to another. At least, we have not found any scientific evidence that supports that claim. Thus, a lot of research can be done yet regarding the use of single-point metaheuristics (other than SA and TS). Related to this, it is possible to observe in the reviewed literature a trend to develop hybrid algorithms,

which combine different types of metaheuristics as well as metaheuristics with machine learning and statistical techniques. While hybridization can be an effective strategy to solve complex problems, it might also add some degree of additional complexity to the solving algorithms. This, in turn, might make them more difficult to be clearly understood, correctly implemented, and applied in practical scenarios. Adding complexity to algorithms –e.g., additional parameters that require fine tuning– also makes it difficult to reproduce their experimental results. For those reasons, only in cases in which significant improvements in performance are obtained, is the hybridization of algorithms justified.

With regards to data, recent research has shown a trend to employ different risk measures to more accurately depict the characteristics of the underlying data. This is also a particularly interesting research area as further stakeholders of financial optimizations (e.g. SMEs) do not provide traditional optimization inputs (financial data) and thus alternative measurements of risk are promising. Further concerning measuring, while hybridizations of simulation and optimization have recently been developed and gained popularity in the application to SCOPs in different application areas, the above finance-related problems have not yet been extensively addressed by simheuristics, even though financial data is characterised by stochastic macro- as well as firm-level uncertainty. It can thus be expected that this research line is promising, with respect to both, the design of new problems at the interface of the two main research areas that have been treated separately in the literature but are fairly interrelated and the application of simheuristics to established COPs that previously neglected stochastic uncertainty on the one hand and the newly formulated COPs on the other.

8.3 The POP

The single-objective POP has been already introduced in the previous section. Here, a richer version is addressed which includes realistic constraints. A binary variable z_i $\forall i \in \{1, 2, \ldots, N\}$ is created to reveal if an asset i is selected ($z_i = 1$ if $w_i > 0$) or not ($z_i = 0$ otherwise). The number of assets in the portfolio, $\sum_{i=1}^{N} z_i$, is bounded by user-defined values, k_{min} and k_{max} (cardinality constraints). Moreover, the user can pre-select certain assets to be included in the portfolio, i.e.: $\forall i \in \{1, 2, \ldots, N\}$, $p_i = 1$ if the asset i is pre-assigned (i.e., $w_i > 0$) and $p_i = 0$ otherwise (pre-assignment constraints). Finally, for each asset i, its associated quantity in the portfolio, w_i, is bounded by user-defined values, ε_i and δ_i (quantity constraints).

The complete mathematical model is:

$$Min \sum_{i=1}^{N} \sum_{j=1}^{N} w_i w_j \sigma_{ij}, \tag{8.22}$$

$$\sum_{i=1}^{N} w_i \mu_i \geq R \tag{8.23}$$

$$\sum_{i=1}^{N} w_i = 1 \tag{8.24}$$

$$0 \leq w_i \leq 1, \forall i \in \{1, 2, \ldots, N\} \tag{8.25}$$

$$k_{min} \leq \sum_{i=1}^{N} z_i \leq k_{max} \tag{8.26}$$

$$\varepsilon_i z_i \leq w_i \leq \delta_i z_i, \forall i \in \{1, 2, \ldots, N\} \tag{8.27}$$

$$0 \leq \varepsilon_i \leq \delta_i \leq 1, \forall i \in \{1, 2, \ldots, N\} \tag{8.28}$$

$$p_i \leq z_i, \forall i \in \{1, 2, \ldots, N\} \tag{8.29}$$

$$z_i \leq Mw_i, \forall i \in \{1, 2, \ldots, N\} \tag{8.30}$$

$$z_i \in \{0, 1\}, \forall i \in \{1, 2, \ldots, N\} \tag{8.31}$$

Equations (8.22) – (8.25) outline the unconstrained optimization problem and determine the unconstrained efficient frontier (UEF). Specifically, Equation (8.23) provides the lower bound for the investor's required return. Equation (8.24) ensures that portfolio weights add up to unity. The purpose of Equation (8.25) is to regulate leveraged positions. By solving the constrained optimization problem given by Equations (8.22) – (8.31) the CEF is obtained. Equation (8.26) formulates cardinality constraints. Equation (8.27) defines quantity constraints. A minimum and maximum quantities of wealth invested in asset i is given by ε_i and δ_i. Both parameters ε_i and δ_i range from 0 to 1 (Equation (8.28)). Given a vector of N binary decision variables Z (Equation (8.31)) (where z_i takes on value 1 if included in the portfolio and 0 otherwise), and a binary vector P of pre-assignments (in which p_i takes on value 1 if pre-assigned and 0 otherwise), whenever asset i is pre-assigned, it has to be included in the portfolio (Equation (8.29)). In Equation (8.30), M is a large positive value such that $Mw_i \geq 1$ for all $w_i \geq 0$. Thus, if the quantity in the portfolio of asset i, w_i, is equal to 0, it means that this asset is not included in the portfolio (i.e., $z_i = 0$).

According to the problem description, the output of the algorithm will be an assets-investment plan, $W = (w_1, w_2, \ldots, w_N)$, satisfying all the aforementioned constraints and with the lowest possible risk. In order to perform a fair comparison with some previous works and existing benchmarks (Chang et al., 2000; Schaerf, 2002; Armañanzas and Lozano, 2005; Moral-Escudero et al., 2006; Fernández and Gómez, 2007), in this paper we will not consider pre-selected assets. However, due to its flexibility, the ARPO algorithm could be adapted without too much effort to deal with this constraint too.

8.3.1 Methodology

The ARPO matheuristic combines three main components: *(i)* an ILS framework; *(ii)* the use of a biased randomization process that guides the generation of new 'promising' solutions (perturbation stage); and *(iii)* the use of a quadratic programming solver that, given a current portfolio, optimizes the levels of investment of each asset (local search). Pseudo-code 4 shows the main procedure of the ARPO algorithm. Apart from the inputs defining the instance, also the maximum computing time allowed, *maxTime*, and an additional parameter, *beta*, are passed to the procedure –the use of this additional parameter will be discussed later.

The ARPO procedure starts by generating a 'dummy' initial solution. This initial solution is constructed by including the assets with the highest return levels so that it provides the highest possible expected return while satisfying all the remaining constraints. This way, if the expected return provided by this solution does not reach the minimum return threshold imposed by the investor, then the problem will be infeasible since no other solution will do it. Notice, however, that it is also likely to obtain a high risk associated with this initial solution.

At this point, a quick local search is applied to this initial solution, which uses quadratic programming in order to optimize the investment level assigned to each asset in the current solution. The improved solution will be considered both as the current 'base' solution and the 'best-so-far' solution. Now, the ARPO procedure resumes by starting an iterative improvement process. It comprises three stages: *(i)* the *perturbation* stage, which applies strong changes to the current base solution in order to increase exploration of the space of solutions; *(ii)* the *local search* stage, which tries to perform a quick improvement of the current base solution by applying some operators –in our case, it is based on the combined use of quadratic programming and a cache memory; and *(iii)* the *acceptation* stage, which in our case makes use of a credit-based system in order to allow accepting, under certain restrictive conditions, a new base solution even when it offers a slightly higher risk than the current base solution.

As regards as the *perturbation* stage (Pseudo-code 5), this follows a destruction - reconstruction process. First, this process takes as an input the current base solution. Second, the current base solution is partially destroyed according to some random criterion –in our case, a randomly selected number of assets are deleted from the portfolio. Third, the destroyed solution is reconstructed (completed) by adding new assets to the portfolio.

Algorithm 4 Main procedure of the ARPO algorithm.

```
    procedure ARPO(inputs, minReturn, maxTime, beta)
 1: initSol ← genInitSol(inputs)                    ▷ generate sol with highest possible return rate
 2: if {getReturn(initSol) < minReturn} then
 3:     return unfeasible                                                    ▷ unfeasible problem
 4: end if
 5: genFriendshipLists(inputs)                  ▷ generate a sorted list of "friends" for each asset
 6: baseSol ← QPOptimize(initSol, minReturn)      ▷ optimize levels for each asset in portf.
 7: baseSol ← cleanSol(baseSol)                   ▷ delete from portf. assets with level = 0
 8: bestSol ← baseSol                                                    ▷ initialize bestSol
 9: elapsedTime ← 0
10: credit ← 0                                          ▷ used in the acceptance criterion
11: while {elapsedTime < maxTime} do                                     ▷ iterated local search
12:     newSol ← perturbateSol(baseSol, inputs, beta)    ▷ destruction-construction stages
13:     if {getMaxReturnAsset(newSol) < minReturn} then          ▷ fix solution if unfeasible
14:         newSol ← repairSol(newSol, inputs)
15:     end if
16:     if {newSol is in cache} then                                ▷ already optimized levels
17:         newSol ← loadFromCache(newSol)          ▷ use optimized levels saved in cache
18:     else          ▷ apply a local search based on quadratic programming optimization
19:         newSol ← QPOptimize(newSol, minReturn)   ▷ optimize levels f.e. asset in portf.
20:         newSol ← cleanSol(newSol)                ▷ delete from portf. assets with level = 0
21:         saveInCache(newSol)
22:     end if
23:     delta ← getRisk(newSol) - getRisk(baseSol)                ▷ newSol improves baseSol
24:     if {delta < 0} then
25:         credit ← -delta
26:         baseSol ← newSol
27:         if {getRisk(newSol) < getRisk(bestSol)} then          ▷ newSol improves bestSol
28:             bestSol ← newSol
29:         end if
30:     else{delta > 0 and delta ≤ credit}                          ▷ acceptance criterion
31:         credit ← 0
32:         baseSol ← newSol
33:     end if
34:     update elapsedTime
35: end while
36: return bestSol
    end procedure
```

Algorithm 5 Perturbation procedure to generate new 'promising' solutions.

```
    procedure perturbateSol(baseSol, inputs, beta)
 1: newSol ← copySol(baseSol)
                    ▷ 1. Remove a random number of randomly selected assets (destruction stage)
 2: nAssetsInSol ← getNAssetsInSol(newSol)
 3: if {nAssetsInSol > 1} then
 4:     nAssetsToRemove ← genRandomNumber(1, nAssetsInSol - 1)
 5:     for {i = 1 to nAssetsToRemove} do
 6:         asset ← selectRandomAsset(newSol)
 7:         newSol ← removeAsset(asset, newSol)
 8:     end for
 9: end if
                        ▷ 2. Randomly select one asset in current portf. to add several of its "friends"
10: asset ← getRandomAsset(newSol)
          ▷ 3. Use biased rand. to add friendly assets until reaching kMax (re-construction stage)
11: while {size(newSol) < getKMax(inputs)} do
12:     listOfFriendlyAssets ← getFriendlyList(asset)          ▷ Sorted list of friendly assets
13:     do          ▷ Randomly select a position using a Geometric(beta) prob. distribution
14:         position ← biasedRandom(size(listOfFriendlyAssets), beta)
15:         newAsset ← getAsset(listOfFriendlyAssets, position)
16:     while {newAsset in newSol}          ▷ Repeat until newAsset not in current portf.
17:         newSol ← addAsset(newAsset, newSol)
18:         asset ← newAsset
19: end while
20: return newSol
    end procedure
```

During this re-construction process, the selection of each new asset added to the portfolio is done following a 'friendship' criterion, i.e.: although the selection of the new asset is random, this new asset will be most likely selected among those assets that are highly compatible –i.e., showing a low covariance value– with the last asset added to the portfolio. This special behavior is attained throughout the use of a biased randomization selection process, which makes use of a geometric distribution of parameter *beta* ($0 < beta < 1$).

Finally, there might be times in which the newly generated solution does not fulfil the minimum return requirement. In those cases, a 'repair' stage is used to swap a randomly selected asset in the current portfolio by a high-return asset not currently in the portfolio (Pseudo-code 6).

Algorithm 6 Repair procedure to make newly generated solutions feasible.

```
    procedure perturbateSol(baseSol, inputs, beta) procedure repairSolution(sol, inputs)
 1: unusedAssets ← getAssetsNotInSol(sol, inputs)          ▷ Consider assets not in portf.
 2: unusedAssets ← shuffle(unusedAssets)          ▷ Random sorting of the unused assets list
 3: assetA ← getRandomAsset(sol)          ▷ Select a random assetA in current portf.
 4: sol ← deleteAsset(assetA, sol)          ▷ Delete assetA from current portf.
 5: for {each asset assetB in unusedAssets} do          ▷ Search unused assetB with high return
 6:     if {getReturn(assetB) ≥ minReturn} then
 7:         sol ← addAsset(assetB, sol)          ▷ Add assetB to current port.
 8:         return sol
 9:     end if
10: end for
11: end procedure
```

8.3.2 Computational experiments

The ARPO algorithm has been implemented as a Java application. We experiment with two sets of stock market data already used in previous studies. The first set was retrieved from the repository ORlib: http://people.brunel.ac.uk/~mastjjb/jeb/orlib/portinfo.html. These instances were proposed by Chang et al. (2000) and were studied by Schaerf (2002), Armañanzas and Lozano (2005), Moral-Escudero et al. (2006), Fernández and Gómez (2007), and Di Gaspero et al. (2011). The data set comprises constituents of five stock market indices, Hang Seng (Hong Kong), DAX 100 (Germany), FTSE 100 (United Kingdom), S&P 100 (United States) and NIKKEI 225 (Japan). These indices were extracted from DataStream and are measured at weekly frequency spanning the period from March 1992 to September 1997.

The portfolio frontier was divided into 100 equidistant points on the vertical axis that represents the user-defined rate of expected portfolio return. Although the algorithm has been designed for the constrained case, it is initially tested on the unconstrained mean-variance optimization problem. The test results show that our solver is able to return solutions that are overlapping with the UEF published at the OR Library, which contributes to validate the effectiveness of our approach.

Next, the algorithm was executed on a constrained mean-variance frontier (the algorithm is executed 30 times and both the best and average results are recorded). The maximum time of execution for each instance is 20 seconds. The benchmark constraints are those imposed by the previous authors. The constraints involve the following conditions: $\varepsilon_i = 0.01$, $\delta_i = 1$, $k_{min} = 1$, $k_{max} = 10$, $\forall i \in \{1, 2, \ldots, n\}$. As in the aforementioned studies, pre-assignment constraints are not considered. Notice that, despite other authors claim that their approaches can solve the constrained problem with all the aforementioned constraints, this fact is not clearly showed, since the parameter values they use do not seem to impose a real challenge for their algorithms in terms of tight constraints.

8.3.3 Analysis of results

Table 8.4 shows the values of average percentage loss (APL) and associated computational times. Notice that, in terms of the minimum APL, our ARPO algorithm outperforms on Instances 2 – 5 the hybrid solvers proposed by Di Gaspero et al. (2011), which comprise combinations of first descent and steepest descent with quadratic programming (FD+QP and SD+QP, respectively). With regard to the first instance, our APL is greater, but this result may emanate from rounding errors. In terms of computational time, ARPO shows a superior performance relative to that of the solver's SD+QP and is comparable or better than the solver's FP+QP performance. We next contrast our results with the results reported by Schaerf (2002) and Moral-Escudero et al. (2006). Although the minimum APL provided by ARPO is slightly superior to the hybrid solver combining a GA and QP in Moral-Escudero et al. (2006), on the remaining instances the minimum APL accomplished by ARPO is lower. Furthermore, our computational times are considerably lower than those reported by the TS in Schaerf (2002), and by GA+QP in Moral-Escudero et al. (2006).

The UEF (as provided in the ORlib) and CEF (as provided by ARPO) for the five stock market indices are compared in Panels A – E of Figure 8.4.

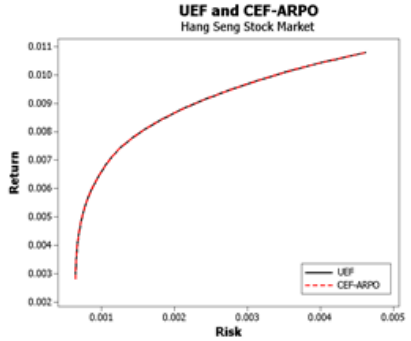
Panel A depicts the CEF for the Hang Seng stock market. A visual inspection suggests that for the Hang Seng stock index the CEF is hardly distinguishable from the UEF. However, as the rate of expected return increases, along with increasing risk of investment, the CEF tends to diverge relatively less from the UEF. In particular, at the higher end of the CEF that features rewarding but risky portfolios, the expected rate of return can be attained with fewer assets.

Panel B depicts the CEF for the DAX 100 stock market. Visual inspection indicates that for the DAX 100 stock index the CEF diverges from the UEF at the lower end of expected return, more specifically, for $R < 0.006$. As the rate of expected return increases, the CEF becomes indistinguishable from the UEF. Notably, portfolios with an expected return at the lower end of the CEF tend to be riskier (i.e., with higher portfolio variance) than portfolios on the UEF.

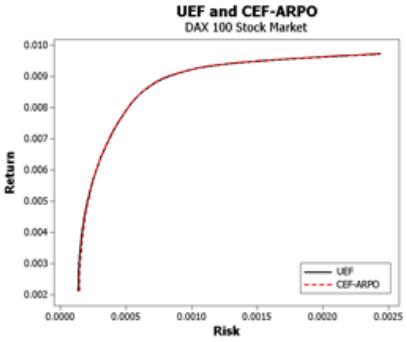
Panel C depicts the CEF for the FTSE 100 stock market. It indicates that for the FTSE 100 stock index –similarly to the DAX 100 stock index– the CEF departs from the UEF at the lower end of expected return. As the rate of expected return increases, the CEF converges to the UEF. Noteworthy, portfolios featuring an expected return at the lower end of the CEF tend to be riskier (i.e., with higher portfolio variance) than portfolios on the UEF. At the higher end of the CEF

TABLE 8.4: Summary of results.

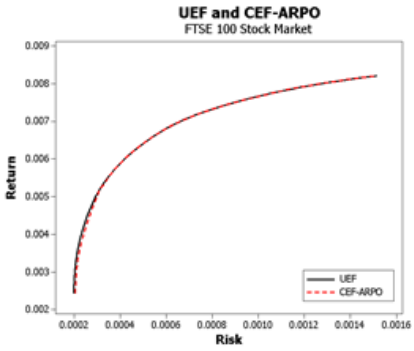
Instance	Di Gaspero et al. (2011)				Moral-Escudero et al. (2006)		Schaerf (2002)		ARPO	
	FD + QP		SD + QP		GA + QP		TS		LS + QP	
	APL	T(s)	APL	T(s)	APL	T(s)	APL	T(s)	APL	T(Std) (s)
HS	0.00366	1.5	0.00321	3.1	0.00321	415.1	0.00409	251	0.00399	2.0 (0.87)
DAX 100	2.66104	9.6	2.53139	14.1	2.53180	552.7	2.53617	531	2.45403	1.0 (1.0)
FTSE 100	2.00146	10.1	1.92146	16.1	1.92150	886.3	1.92597	583	1.88340	13.7 (9.28)
S&P 100	4.77157	11.2	4.69371	18.8	4.69507	1163.7	4.69507	713	4.65095	15.5 (8.42)
NIKKEI 225	0.24176	25.3	0.20219	45.9	0.20198	1465.8	0.20198	1603	0.20189	5.0 (17.20)



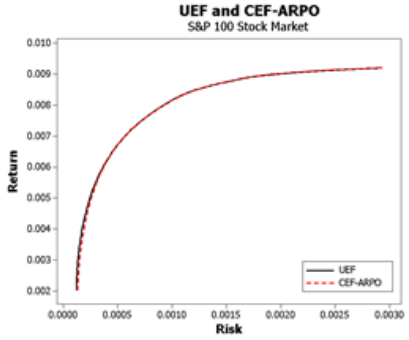
Panel A – Hang Seng Stock Market



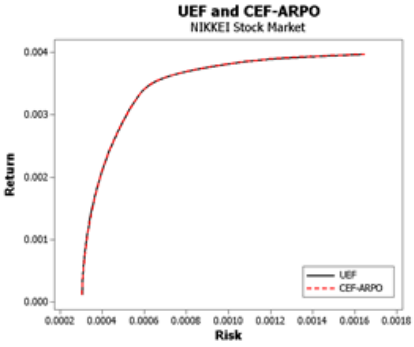
Panel B – DAX 100 Stock Market



Panel C – FTSE 100 Stock Market



Panel D – S&P 100 Stock Market



Panel E – NIKKEI Stock Market

FIGURE 8.4: UEF and CEF-ARPO.

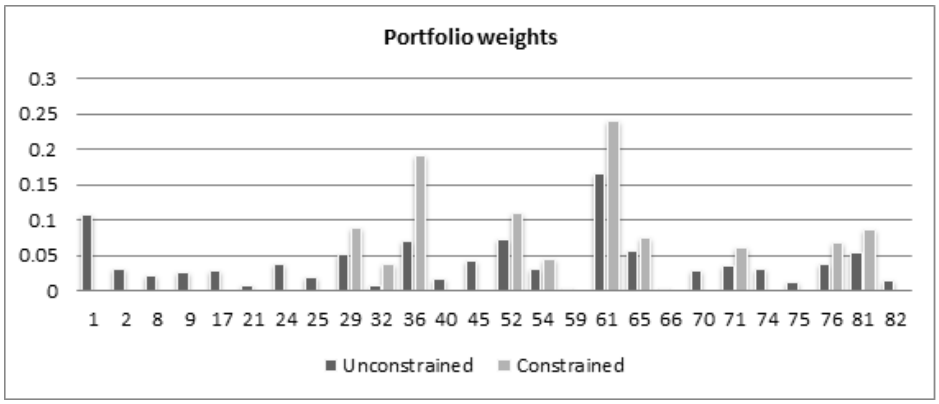


FIGURE 8.5: Portfolio weights for the FTSE 100 stock indices.

that features rewarding but risky portfolios, the expected rate of return can be achieved with fewer assets.

Panel D depicts the CEF for the S&P 500 stock market. It indicates that for the S&P 500 stock index –as with the DAX 100 and FTSE 100 stock indices– the CEF departs from the UEF at the lower end of expected return. As the rate of expected return increases, the CEF becomes visually indistinguishable from the UEF. Portfolio investments with an expected return at the lower end of the CEF involve relatively more risk than portfolios with the same expected return located on the UEF.

Finally, Panel E depicts the CEF for the NIKKEI stock market. It indicates that for the NIKKEI stock index, the relation between the CEF and the UEF follows a pattern similar to the Hang Seng stock index. Specifically, although the CEF departs from the UEF at the lower end of expected return, the difference is visually very small. As the rate of expected return increases, the CEF gradually approaches the UEF. At the higher end of the CEF that includes portfolios with high expected return and high risk, the APL approaches to zero.

To evaluate differences between the UEF and the CEF-ARPO, we also provide the portfolio weights for Instance 3 (FTSE 100) (Figure 8.5), where the required rate of return is 0.0041572635, which is an approximately central value within the overall of returns. This instance was executed twice with the same seed. The UEF considered all assets with weights ranging from 0 to 1 inclusively. The CEF was constrained to the minimum of 1 and the maximum of 10 assets, with portfolio weights ranging from 0.01 to 1. The minimum values of the portfolio variance were 2.3872556507357437E-4 (the UEF) and 2.5098945345432527E-4 (CEF-ARPO). The percentage loss is 5.137%. The UEF selected 26 assets, whereas the CEF portfolio selected 10 assets. The 10 assets are the subset of assets selected in the UEF portfolio.

8.4 The SPOP

The difference between the POP and the SPOP considered lies in the modelling of asset returns and covariances. While they are represented by expected values in the first case, the second considers realistic stochastic uncertainty and thus treats these as random variables. This results in a modified return constraint where a return no lower than R must be achieved with a probability of, at least, P_0.

The mathematical formulation for the SPOP requires two modifications:

- Covariances (C_{ij}) in the objective function are considered to be random variables following a given probability distribution (e.g., the one that best fits the historical data available):

$$f(x) = \sum_{i=1}^{n} \sum_{j=1}^{n} w_i w_j C_{ij} \geq R \tag{8.32}$$

- Equation 8.23 is replaced by the following probabilistic constraint:

$$P(\sum_{i=1}^{N} R_i w_i \geq R) \geq P_0 \tag{8.33}$$

where R_i refers to the asset return modeled as a random variable. It ensures that the portfolio return will be no lower than the threshold R with a probability of, at least, P_0.

8.4.1 Methodology

The VNS metaheuristic is proposed as a base framework. The methodology includes biased randomization techniques and employs the open-source quadratic programming solver ojAlgo (http://ojalgo.org), developed in Java, to determine the weights allocated to a given set of assets. Additionally, a cache memory (implemented as a hash map data structure) is used in order to avoid calling the solver repeatedly for a specific set of assets.

The flowchart diagram of our approach is depicted in Figure 8.6 and described next:

1. Consider a SPOP instance defined by N assets. Each asset i has an associated return rate R_i, which is a random variable following a probability distribution, either empirical or theoretical. Each pair of assets i, j is characterised by a covariance C_{ij}, which is also random and depends on the correlation P_{ij} and the standard deviations S_i and S_j according to the following equation: $P_{ij} = \frac{C_i j}{S_i S_j}$

2. Transform the original stochastic problem into a POP instance by means of replacing the random variables by their expected values μ_i and $\sigma_i j$.

3. Construct an initial solution ($initSol$) by selecting the k_{min} assets with the highest returns, after including the s assets pre-selected by the investor, and calling the solver. Afterwards, simulation techniques are considered to compute the probability of satisfying the return constraint in the stochastic environment described by the original instance. In particular, a short number of scenarios (sim_{short}) is used to simulate returns. The solution is stored and one moves on to the fourth step, provided the constraint is satisfied. If this is not the case, a feasible solution is searched using a randomised and iterative procedure. First, the pre-selected assets compose a portfolio. In the next step, the non-preselected assets are ordered according to their expected return, and a random number, between $k_{min} - s$ and $k_{max} - s$, are selected using biased randomization, relying on a geometric distribution with a parameter β (Juan et al., 2011b). All weights are set to the minimum value initially, and then, each weight is set to the maximum value possible (taking into account for an asset a_i the following elements: ε_i, δ_i, and the fraction that remains to be allocated, i.e., $1 - \sum_{i=1}^{n} x_i$) in the order previously established. If an initial solution can be constructed through this, one moves to step 4. It is worthwhile to remark that we avoid using the solver at this step because the focus is on finding an initial feasible solution considering the stochastic environment and not the one with the lowest risk. The time spent searching for a feasible solution is limited by T_{init}, and the algorithm execution stops if no feasible solution is obtained.

4. A list $bestSols$ is created for storing the l best-found solutions in terms of expected risk. Then, $initSol$ is copied into $currentSol$ and k is set to one. Following this, the expected risk of $currentSol$ is computed by using MCS, and the solution is included in the created $bestSols$ list.

5. An iterative procedure is started and steps 6 and 7 are executed during a given amount of time (T_{loop}).

6. A new solution ($newSol$) is created by shaking the current one. This procedure consists of randomly erasing a number of non pre-selected assets in the solution and randomly introducing new assets until reaching k_{max}. The number of assets erased is determined by k. Moreover, a local search is applied to the resulting solution. It aims to improve the solution by replacing the asset with the lowest weight with another one from the list of non-selected assets ordered

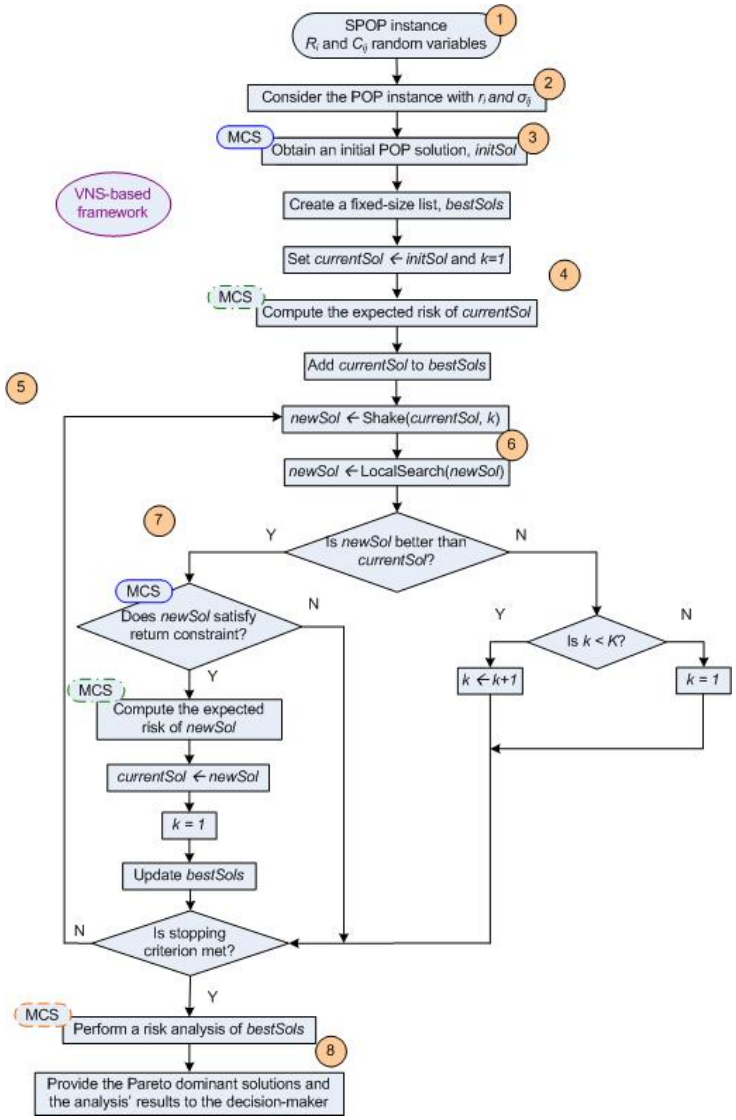


FIGURE 8.6: Flowchart of the proposed approach for the SPOP.

7. *newSol* is compared against *currentSol*. If the former is better in terms of risk associated with the deterministic version of the problem, *newSol* is considered to be a promising portfolio setting and the return constraint for the stochastic environment is checked for it. In case of being satisfied, the expected risk is computed for the stochastic version of the problem. If the expected risk of *newSol* is better than that of *currentSol*, then *newSol* replaces *currentSol*, k is set to one and *bestSols* is updated. If it is not satisfied, the solution is discarded. If *newSol* is not better, k is increased in one unit if $k < K$ or set to one otherwise.

8. Once the iterative procedure ends, the algorithm returns *bestSols*. For each solution, a sample of risk measurements is obtained by simulating a large number of scenarios (sim_{large}). We perform a risk analysis where solutions are compared using the distributions of risk. In order to simplify the analysis, it is based on the expected values and the variances of the distributions, and the reliabilities (or probabilities of satisfying the return constraint). Accordingly, the Pareto dominant solutions (i.e., those that are not dominated by another portfolio for one measure while the other measures are at least equally good) are reported to the decision-maker.

8.4.2 Computational experiments

The described algorithm has been implemented as a Java application. The algorithm is executed ten times using different seeds; only the best results are stored. Stock market data from the repository ORlib (http://people.brunel.ac.uk/~mastjjb/jeb/orlib/portinfo.html) is used. This set was presented by Chang et al. (2000), and has been largely analyzed (Schaerf, 2002; Armañanzas and Lozano, 2005; Moral-Escudero et al., 2006; Fernández and Gómez, 2007; Di Gaspero et al., 2011). This benchmark instance is deterministic. It has been adapted by replacing the deterministic returns and covariances by random variables. More specifically, the following complementary scenarios have been considered:

- S_i (Standard deviation) follows a $LN(\mu_S, \sigma_S)$, where LN represents a Log-Normal distribution, and μ_S and σ_S are the mean and the standard deviation of the variable natural logarithm, respectively. They may be determined by the value of the mean and the standard deviation of the variable that are set to σ_i and $c\sigma_i$, being c an input.
- $P_i j$ (Correlation) follows a $TN(\mu_P, \sigma_P, l, u)$, referring TN to truncated Normal distribution, where the parameters are the mean, the standard deviation, and the lower and upper limit, respectively. μ_P is set to the original correlation $\rho_i j$, while σ_P is an input. By the definition of correlation, l and u are set to -1 and 1, respectively. A special case is when $i = j$, then l and u are equal to 1 (i.e., $P_{ij} = 1$).
- R_i follows a $N(\mu_R, \sigma_R)$, where μ_R and σ_R are the mean and the standard deviation of the variable, respectively, which may be determined by the value of the mean and the standard deviation of the variable that are set to r_i and S_i, respectively.

Three values for $c(0.01, 0.025, 0.08)$ and $\sigma_P(\sqrt{0.00002}, \sqrt{0.0002}, \sqrt{0.002})$ have been tested in order to explore three different levels of stochasticity. The former values have been selected after performing some quick tests to explore the "reasonable" range for each parameter. Two computational experiments have been carried out. The first experiment considers stochastic covariances (first two scenarios). The second experiment builds on the first one, but additionally introduces stochastic returns (all three scenarios).

The parameter fine-tuning has been performed taking into account suggestions of other authors and results from fast experimental tests. The recommended number of neighbours (K) is 3 (Hansen et al., 2010a). A movement in each neighbour involves changing 25%, 35%, and 45% of the assets, respectively. Regarding the number of solutions stored to analyse at the end (l), a total of 10 are considered. As suggested in Juan et al. (2011b), β is randomly selected from a uniform distribution with parameters 0.05 and 0.25. Finally, sim_{short} and sim_{large} are set to 2500 and 12500, respectively, T_{init} and T_{loop} are set to 5 and 15, respectively.

8.4.3 Analysis of results

Table 8.5 summarises the results of the first experiment. The first experiment compares two types of solutions: (*i*) the best-found solution to the deterministic version of the problem (BDS); and (*ii*) the best-found solution to the stochastic version (BSS). Different levels of variability (variance) in the random variables modelling covariances and returns are considered. For each of these variability levels (low, medium, and high) a different stochastic scenario is defined. Notice that portfolio configurations obtained for the deterministic version of the problem can also be used as investment plans for the stochastic version of the problem. Different risk measures (costs) associated with the BDS portfolio configuration are considered: the risk measure obtained when employing the BDS in a deterministic scenario, and the expected risk value obtained when using it in each stochastic scenario. The former could be considered as a lower bound for the BSS, while the latter could be considered as an upper bound for the BSS. The first column reveals the required return. The next four columns depict the BDS. The following three columns contain the expected risk associated with the BSS for each of the stochastic environments analysed. Also, the average computational time needed for finding the BSS is provided. The last five columns gather some gaps: (*i*) the gap between the risk and the expected risk for a low level of stochasticity of the BDS; (*ii*) the gap between the risk of the BDS and the expected risk of the BSS (also for a low level of stochasticity); (*iii*) the gap between the expected risks for the BDS and the BSS considering the environment of a low risk, which quantifies the benefit of using the simheuristic

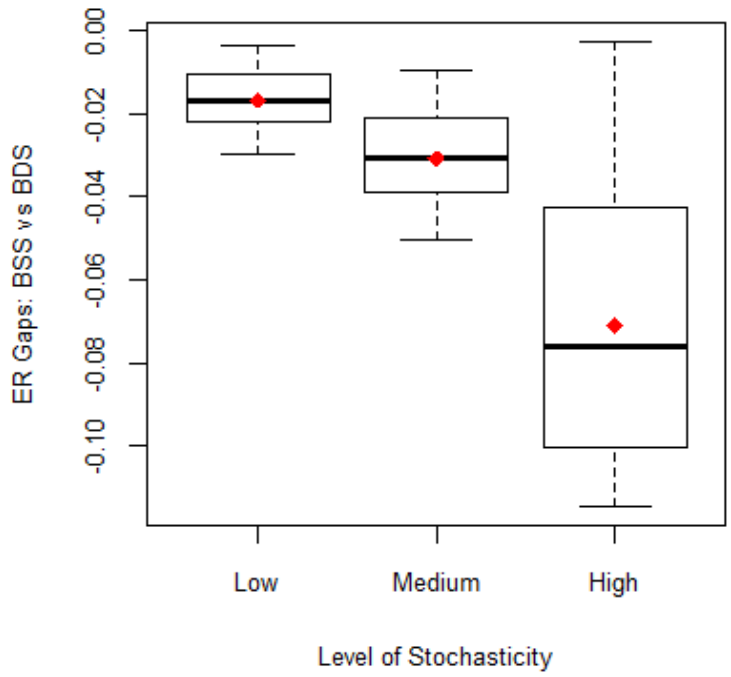


FIGURE 8.7: Risk gaps between the best deterministic and stochastic solutions for different levels of stochasticity (environments).

approach instead of assuming constant values; and (*iv*) the previous gap considering the other two environments. Additionally, the average of each ratio has been added at the bottom of the table. Figure 8.7 illustrates boxplots of the gaps regarding expected risk between the BDSs and BSSs for the different environments. The expected risk of the BDS is subtracted from that of the BSS so that negative gaps indicate an improvement in the expected risk of the solution. Mean values are represented by diamonds.

Based on these outputs, it may be concluded that our algorithm is able to obtain a reasonably good BSS in 0.73 seconds on the average. The gaps between the risk of the BDS and the expected risk of the BDS (when used as a portfolio configuration for the stochastic environment) and the BSS are quite high even for the low-variability scenario (11.82% and 9.92% on the average). As expected, the measure of the BSS is closer to the lower-bound (the risk) than the one of the BDS. Regarding the benefits of using the simheuristic approach in comparison with assuming constant values in terms of expected risk, the mean gaps found for each environment are: -1.68%, -3.09%, and -7.10%. It is important to remark that the gaps are never positive. Thus, the BSS shows a lower expected portfolio variance than the BDS when the latter is used to solve the stochastic version of the problem. Furthermore, the performance of the BDS deteriorates when covariances become more uncertain.

Results from the second experiment are displayed in Table B.2. As in the previous table, the first column identifies the required return. Columns 2, 3, and 4 detail the expected risk of the BSSs under the lower, medium, and high levels of uncertainty, given a probability of 50% for attaining the required rate of return. Columns 5, 6, and 7 report the gaps between the expected values of risk of the BSSs under the lower, medium, and high levels of uncertainty, when the probabilities of attaining the required return are 50% and 47%, respectively (since the benchmarks used are extensions of classical ones for the deterministic version, only some probability values make sense). Finally, the average computational time is provided.

It can be concluded that the gap between the expected risk of the BSS requiring a probability of 47% and the one with a probability of 50% is relatively small, although it can be relevant and high in some cases. The average values for the different environments are: 0.31%, 0.39%, and 1.84%. Therefore, the gap increases as the level of stochasticity gets higher.

TABLE 8.5: Hang Seng Stock Market (Hong Kong) with stochastic covariances.

		Deterministic Solution			Stochastic Solutions			Time (s.)	Gaps (%)				
Required Return	Risk (1)	E.R. - Low (2)	E.R. - Med. (3)	E. R. - High (4)	E.R. - Low (5)	E.R. - Med (6)	E.R. - High (7)		(2-1)	(5-1)	(5-2)	(6-3)	(7-4)
0.002861137	0.0006424	0.0007055	0.0007975	0.001114	0.0006948	0.000777	0.0010713	0.211	0.0982	0.0815	-0.0152	-0.0257	-0.0383
0.002941981	0.0006429	0.0007062	0.0008052	0.001195	0.0006952	0.0007772	0.0010701	0.409	0.0985	0.0813	-0.0156	-0.0348	-0.1045
0.003022827	0.0006437	0.0007015	0.0007888	0.0011057	0.0006961	0.0007782	0.0010706	0.627	0.0897	0.0814	-0.0076	-0.0135	-0.0318
0.003103671	0.0006444	0.0007108	0.0008089	0.0011647	0.0006976	0.0007799	0.0010727	0.456	0.1031	0.0826	-0.0186	-0.0358	-0.079
0.003184516	0.0006455	0.0007054	0.0007978	0.00113	0.0006996	0.0007824	0.0010764	0.369	0.0929	0.0838	-0.0083	-0.0192	-0.0474
0.003265361	0.0006468	0.0007057	0.0007951	0.0011208	0.0007011	0.000785	0.0010821	2.03	0.0912	0.084	-0.0065	-0.0127	-0.0346
0.003346206	0.0006483	0.0007113	0.0008062	0.0011714	0.0007021	0.0007857	0.0010806	2.461	0.0972	0.083	-0.0129	-0.0255	-0.0775
0.003427051	0.00065	0.0007143	0.0008095	0.0011649	0.0007035	0.0007869	0.0010809	1.541	0.0989	0.0824	-0.015	-0.0279	-0.0721
0.003507896	0.0006517	0.0007161	0.0008037	0.0010852	0.0007103	0.0007887	0.0010822	5.966	0.0989	0.09	-0.0081	-0.0187	-0.0028
0.00358874	0.0006536	0.0007103	0.000799	0.0011332	0.0007076	0.0007912	0.0010848	0.581	0.0867	0.0826	-0.0038	-0.0098	-0.0427
0.010137479	0.0035774	0.0040554	0.0047712	0.0074653	0.0039666	0.0045855	0.0067181	0.001	0.1336	0.1088	-0.0219	-0.0389	-0.1001
0.010218315	0.0036908	0.004178	0.0049221	0.0076114	0.0040965	0.0047415	0.0069622	0.002	0.132	0.1099	-0.0195	-0.0367	-0.0853
0.010299151	0.0038091	0.0043348	0.0051217	0.0079629	0.0042323	0.0049052	0.0072194	0.001	0.138	0.1111	-0.0236	-0.0423	-0.0934
0.010379986	0.0039324	0.004503	0.0053405	0.0083909	0.0043741	0.0050764	0.0074899	0.001	0.1451	0.1123	-0.0286	-0.0494	-0.1074
0.010460822	0.0040605	0.0046604	0.0055344	0.0087777	0.0045219	0.0052551	0.0077734	0.001	0.1477	0.1136	-0.0297	-0.0505	-0.1144
0.010541657	0.0041937	0.004762	0.0055848	0.0083847	0.0046757	0.0054415	0.0080701	0.001	0.1355	0.1149	-0.0181	-0.0257	-0.0375
0.010622493	0.0043317	0.0048879	0.005757	0.0089222	0.0048354	0.0056353	0.00838	0.001	0.1284	0.1163	-0.0107	-0.0211	-0.0608
0.010703329	0.0044747	0.0051485	0.0061357	0.0096893	0.0050011	0.0058368	0.008703	0.001	0.1506	0.1176	-0.0286	-0.0487	-0.1018
0.010784164	0.0046226	0.0052806	0.006288	0.010186	0.0051728	0.0060458	0.0090391	0.001	0.1423	0.119	-0.0204	-0.0385	-0.1126
0.010865	0.0047755	0.0055134	0.0065833	0.010458	0.005381	0.0063068	0.0096622	0	0.1545	0.1268	-0.024	-0.042	-0.0761
							Average	0.733	0.1182	0.0992	-0.0168	-0.0309	-0.071

TABLE 8.6: Hang Seng Stock Market (Hong Kong) with stochastic covariances and correlations.

Required Return	ER (50%)			ER Gaps [%] (50-47%)			Time (s)
	Low	Medium	High	Low	Medium	High	
0.002861137	0.0007006	0.0007872	0.0011358	0.00%	0.00%	4.38%	1.244
0.002941981	0.0007006	0.0007873	0.001135	0.00%	0.00%	4.44%	2.749
0.003022827	0.0007019	0.0007882	0.0011272	0.00%	0.00%	3.66%	2.647
0.003103671	0.0007027	0.00079	0.0011392	0.00%	0.00%	3.24%	1.7
0.003184516	0.0007068	0.0007925	0.0011075	0.33%	0.00%	1.23%	2.715
0.003265361	0.0007093	0.0007954	0.0011243	0.66%	0.00%	2.15%	1.438
0.003346206	0.0007085	0.000796	0.001137	0.09%	0.19%	2.67%	0.633
0.003427051	0.0007085	0.0007983	0.0011017	-0.11%	0.14%	0.00%	0.764
0.003507896	0.0007138	0.000799	0.0011144	0.54%	0.02%	0.05%	1.365
0.00358874	0.0007161	0.0008014	0.0011068	0.39%	0.00%	0.00%	1.726
0.010137479	0.0040004	0.0046595	0.0071471	0.00%	0.00%	1.05%	1.631
0.010218315	0.0041312	0.0048569	0.0074155	0.00%	0.82%	1.15%	2.8
0.010299151	0.004268	0.0049834	0.0076973	0.00%	0.00%	1.23%	1.421
0.010379986	0.0044359	0.0052031	0.0079926	0.57%	0.90%	1.31%	2.937
0.010460822	0.0045867	0.0053878	0.0083013	0.59%	0.93%	1.37%	4.313
0.010541657	0.0047435	0.0055803	0.0086235	0.61%	0.97%	1.42%	2.31
0.010622493	0.0049063	0.0057805	0.0089592	0.63%	1.00%	1.46%	3.76
0.010703329	0.0050752	0.0059884	0.0093083	0.65%	1.02%	1.49%	1.836
0.010784164	0.0052501	0.006204	0.0096709	0.67%	1.05%	1.52%	0.773
0.010865	0.0054126	0.0063772	0.0098998	0.37%	0.70%	3.06%	1.729
			Average	0.30%	0.39%	1.84%	

8.5 Example with stocks and individual commodity futures contracts

Diversification is best achieved through combining assets with low or negative correlation into an investment portfolio. Due to the increased correlation among individual stocks, diversification possibilities of stock portfolios have become limited. Commodities in general and metals in particular on the contrary have shown to yield low correlation with stocks (Jaffe, 1989; Chua et al., 1990; Hillier et al., 2006; Daskalaki and Skiadopoulos, 2011), especially because macroeconomic shocks tend to impact stock and commodity prices in different directions (Silvennoinen and Thorp, 2013). Particularly concerning inflation, the reaction of commodities and stocks might differ fundamentally. Indeed, while unexpected inflation leads to an increase in the prices of commodities, stocks have generally been found to be an inflation-protected asset class (Hardouvelis, 1987; McQueen and Roley, 1993) or, in case of fluctuation, have even yielded falling prices (Fama, 1981; Amihud, 1996; Bansal et al., 2014). However, investment in physical commodities is characterized by high costs (storage) and additional uncertainty (perishable nature, seasonal cycles of the goods) so that commodity futures are a natural alternative, providing the same diversification benefits without the implied disadvantages to an investor. Bansal et al. (2014) calculate the efficient frontier for an investment portfolio made up of indices of commodity futures and stocks and find it to lie above that for a traditional stock and bond portfolio. This diversification takes place independent of the state of the stock market: Crude oil futures contracts were shown to lead to successful diversification in both upward and downward trending stock markets (Geman and Kharoubi, 2008). Commodities emerge as a significant diversifier of both equity returns and volatility (Brooks and Prokopczuk, 2013). Investment in commodities is also demonstrated to significantly improve investor's expected utility. In this regard, Garrett and Taylor (2001) find that expected-utility-maximizing investors, depending on the degree of risk aversion, should invest 30 to 68% of their wealth in commodities. Unlike in Geman and Kharoubi (2008), this finding is event-dependent and period-specific. In contrast to the above mentioned studies that were conducted from the standpoint of a US-based investor, Belousova and Dorfleitner (2012) show that a euro investor can also accrue diversification benefits from commodity investments. In particular, the authors emphasize that industrial metals, agricultural commodities and livestock contribute to the reduction of investment risk, while precious metals and energy are associated with both lower portfolio risk and higher return. Investments in commodities become especially rewarding when

the general financial climate becomes negative (Chow et al., 1999). In the quest for hedging and 'save haven' vehicles against losses in the sovereign bond market, Agyei-Ampomah et al. (2014) highlight the superiority of industrial metals (aluminium, copper, lead, nickel, tin and zinc) over precious metals (gold, silver, platinum and palladium). Antonakakis and Kizys (2015) underline the information contents of gold, silver and platinum in improving forecast accuracy of returns and volatilities of palladium, crude oil and the EUR/CHF and GBP/USD exchange rates. Prominent among commodities is gold – commonly regarded as a 'safe haven' asset – that provides wealth protection by hedging investments in the stock and foreign exchange markets, even during extreme price movements during periods of turmoil (Pukthuanthong and Roll, 2011; Ciner et al., 2013; Reboredo, 2013). The above results have previously been confirmed by You and Daigler (2012). However, they employ individual stocks and futures contracts rather than stocks and commodity indices. This increases the complexity of the optimization problem. In order to cope with this, they resort to a portfolio optimization software package, which is limited to 120 observations. To circumvent this, a metaheuristic algorithm is applied in this paper that is not only capable of dealing with an extensive number of observations, but also with further constraints.

However, an extensive analysis on the diversification benefits of commodity futures by Cheung and Miu (2010) raises concerns about the universal validity of the above findings, indicating that individual assessments become necessary. Furthermore, the ex-post performance of stock and commodity futures portfolios was found to be inferior to that of a portfolio made up of traditional assets by Daskalaki and Skiadopoulos (2011), thus making this another important question in the evaluation. It is thus also evaluated whether the applied methodology is able to identify stable asset weights based on ex-post performance and if so, whether the performance of the portfolio including commodity futures outperforms the traditional stock portfolio.

8.5.1 Problem and data description

There is a set of potential assets to choose from. On the one hand, there is a set of n stocks $S = s_1, s_2, ..., s_n$ and on the other hand, a set of m individual commodity futures contracts $F = f_1, f_2, ..., f_m$ is included, resulting in a total number of potential assets $A = a_1, a_2, ..., a_{m+n}$ equal to $m+n$. For all assets, the expected return based on historical data of a specified time period $E[R_i]$ is calculated. The inclusion of assets with negative expected returns is allowed for two reasons. The first is a technical one: As real-life investors choose from a potential pool of assets whose returns are notably influenced by the historical time horizon chosen for analysis, it prevents introducing a bias. Furthermore, the introduction of futures contracts with slightly negative returns can still cause the portfolio to outperform that composed of only stocks. As a measure of riskiness of the portfolio, its variance is calculated. The mathematical formulation and the methodology very similar to the ones described in Section 3. For this reason, they are not reproduced here.

As the approach is concerned with the comparison of a stocks-alone and a stocks-and-futures portfolio, individual daily historical closing price data for the Dow Jones 30 constituents on the one hand and daily settlement prices for the 21 most actively traded commodity futures prices in the United States covering the period from February 18, 2014 to April 1, 2016 are obtained, resulting in 535 observations for each time series. Due to expiration of fixed-maturity futures contracts, the continuous series are created by data providers by rolling over the futures contracts of different maturities. Table 8.7 presents the average daily returns and the corresponding standard deviations for each of the included stocks and commodity futures contracts.

The average correlations within the two asset classes, as well the mean overall correlation, are presented in Table B.2. Due to the askew distribution of correlations these have been calculated by transforming the individual correlations into Fisher-Z-values, taking the arithmetic mean and then retransforming. It becomes obvious that the correlation between the potential stocks is significantly higher than that within the class of commodity future contracts. Furthermore, the mean correlation between stocks and futures is the lowest overall for the data sample. This reinforces the assumption that a combined portfolio of stocks and futures can lead to superior diversification and a resulting lowering of the associated expected risk for a given portfolio return.

8.5.2 Analysis of results

In the following the results for two experiments are analyzed first with respect to risk analysis of the ex-ante portfolios and then with respect to the ex-post performance, or stability, of the found

TABLE 8.7: Descriptive statistics of stocks and futures.

Assets	Average return	Standard deviation
Stocks	**0.000307%**	**0.012996**
Apple	0.076993%	0.015544
Microsoft	0.084877%	0.015512
Exxon Mobil Corporation	-0.014890%	0.013191
Johnson & Johnson	0.035448%	0.009978
General Electric Company	0.047681%	0.012239
JPMorgan Chase & Co.	0.015270%	0.014037
The Procter & Gamble Company	0.013599%	0.009085
Verizon Communications Inc.	0.032659%	0.009721
Wal-Mart Stores Inc.	-0.010853%	0.011372
Pfizer Inc.	-0.004829%	0.011537
The Coca-Cola Company	0.038688%	0.009100
Chevron Corporation	-0.021950%	0.015983
Visa Inc.	0.069322%	0.014216
The Home Depot, Inc.	0.110242%	0.012426
The Walt Disney Company	0.050015%	0.012805
Merck & Co. Inc.	0.002151%	0.012748
International Business Machines Corporation	-0.026583%	0.012757
Intel Corporation	0.062516%	0.015464
Cisco Systems, Inc.	0.054523%	0.013921
UnitedHealth Group Incorporated	0.116388%	0.014094
McDonald's Corp.	0.058294%	0.010550
3M Company	0.050225%	0.010810
NIKE, Inc.	0.103030%	0.014542
The Boeing Company	0.005434%	0.014169
United Technologies Corporation	-0.017746%	0.011471
The Goldman Sachs Group, Inc.	0.005036%	0.013801
American Express Company	-0.061105%	0.013448
E. I. du Pont de Nemours and Company	0.009985%	0.015360
Caterpillar Inc.	-0.030999%	0.015346
The Travelers Companies, Inc.	0.067270%	0.009718
Commodity futures	**-0.000496%**	**0.000338**
Brentcrudeoil	-0.120960%	0.025457
Copper	-0.043944%	0.012876
Crudeoil	-0.125399%	0.025594
Cocoa	0.017888%	0.012117
Coffee	-0.055508%	0.023227
Corn	-0.031191%	0.014505
Cotton#2	-0.058913%	0.013930
Feedercattle	-0.032845%	0.010810
Gold	-0.004478%	0.009594
Heatingoil	-0.120557%	0.023106
KCWheat	-0.079898%	0.016955
Leanhog	-0.054406%	0.023260
Livecattle	-0.035734%	0.011952
Naturalgas	-0.116111%	0.022118
Orangejuice	-0.003081%	0.020640
Silver	-0.017541%	0.016391
Soybean	-0.047312%	0.015006
Soybeanmeal	-0.030699%	0.020249
Soybeanoil	-0.042044%	0.013330
Sugar#11	0.015679%	0.022557
Wheat	-0.054146%	0.017646

TABLE 8.8: Average correlations between asset classes.

Within stocks	Within futures	Stocks and futures	Overall
0.4385	0.0765	0.0075	0.1756

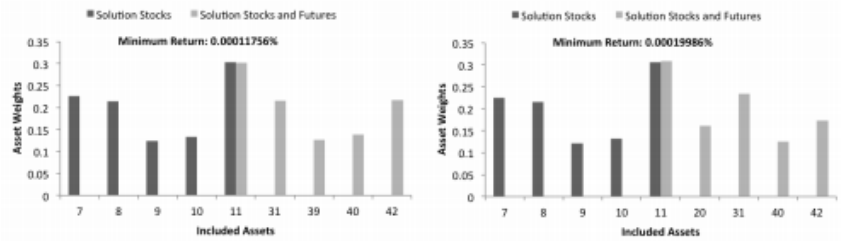


FIGURE 8.8: POP solutions for minimum returns on the lower spectrum.

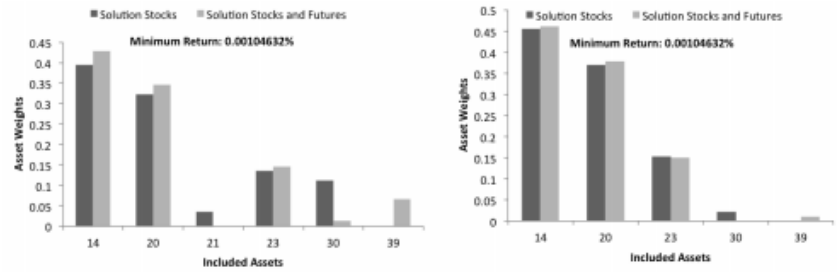


FIGURE 8.9: POP solutions for minimum returns on the higher spectrum.

solutions. Ex-ante portfolios are those portfolios with constituent assets and weights determined by the matheuristic based on the data gathered for the sample period. Ex-post portfolios refer to the application of the ex-ante portfolio asset weights to the data following the sample period at time $t + 1$. It thus refers to a hypothetical investment at time t into the best-found portfolios that is then evaluated one month later at time $t + 1$.

Figure 8.8 and 8.9 present two exemplary solutions. It is to be noted that assets 1 through 30 represent stocks and assets 31 through 51 represent commodity futures contracts. For low-level minimum returns in Figure 8.8, the first stock portfolio contains portions of asset 7, 8, 9, 10, and 11 (all stocks), while the stocks and futures portfolio contains assets 11, 31, 39, 40, and 42 (one stock and four futures contracts). For high-level minimum returns, the solutions are much more similar with respect to the asset composition. The first exemplary stock solution in Figure 8.9 is composed of assets 14, 20, 21, and 30 (all stocks), while the stocks and futures portfolio contains asset 39 instead of asset 21. The exemplary solutions showcase that the solutions for higher minimum returns overlap further and are more similar in terms of selected assets constituents and weights than for low levels of return. This illustrates the finding that the allocation of commodity futures increases with increasing risk aversion of the investor, yielding that they represent a valuable alternative as diversification means especially for lower-risk portfolios.

Risk analysis

Table 8.9 summarizes the results of the two experiments. It essentially compares the results obtained for a particular minimum return. At first sight, the previously mentioned complexity of the problem becomes obvious when the instance times are considered: They significantly increase for the composite portfolios that are selected from an asset pool of 51 potential constituents as opposed to the basic formulation that only considers a pool of 30 stocks. More importantly, the associated risk is presented. As expected, it continuously increases with increasing returns demanded by the investor. The risks between two best-found solutions based on different asset pools are then compared. The existence of a risk gap is explained by the inclusion of individual futures contract in the best-found portfolio compared to the stock-alone portfolios. It is to be noted that this was always done at the expense of excluding at least one previously included stock and not through the addition of new assets in the portfolio, indicating no rise in managerial effort or transaction costs associated with including futures in a traditional stock portfolio. A positive risk gap indicates that the risk was minimized with respect to the solution found for a stock-alone portfolio and thus successfully diversified. This was the case for 94 out of the 99 return instances, while the remainder showed a gap equal to zero. This and the average percentage gap of 26.84% strongly reinforce the initial intuition that individual futures contracts increase diversification beyond that

TABLE 8.9: POP results for a selected subset of minimum returns.

	Stock-alone portfolio		Stock-and-futures portfolio		Gap	Gap [%]
Minimum return	**Risk (1)**	**Time [s]**	**Risk (2)**	**Time [s]**	**(1) - (2)**	**(1) - (2)**
0.0000117564	0.0000531088	0.873	0.0000218480	10.202	0.0000312608	58.86180821%
0.0000822945	0.0000531088	0.042	0.0000230232	0.567	0.0000300856	56.64899226%
0.0001528327	0.0000531088	0.031	0.0000238688	3.141	0.0000292400	55.05678908%
0.0002233709	0.0000533257	0.183	0.0000265600	0.279	0.0000267657	50.19287135%
0.0002939091	0.0000533474	0.058	0.0000271226	12.655	0.0000262247	49.15853444%
0.0003644473	0.0000540647	0.014	0.0000311877	2.541	0.0000228770	42.31411623%
0.0004349855	0.0000546027	0.287	0.0000343818	16.648	0.0000202209	37.03278409%
0.0005055236	0.0000558261	0.225	0.0000375692	8.41	0.0000182569	32.70316214%
0.0005760618	0.0000573681	0.099	0.0000428111	19.99	0.0000145571	25.37472916%
0.0006466000	0.0000601832	1.1	0.0000473472	11.772	0.0000128360	21.32821120%
0.0007171382	0.0000642364	0.254	0.0000543535	17.792	0.0000098829	15.38520216%
0.0007876764	0.0000694180	0.4	0.0000619521	6.255	0.0000074659	10.75499150%
0.0008582145	0.0000766848	3.21	0.0000705611	16.555	0.0000061238	7.98554603%
0.0009287527	0.0000855374	0.091	0.0000812718	13.253	0.0000042656	4.98682448%
0.0009992909	0.0000965467	1.369	0.0000936730	5.567	0.0000028737	2.97648703%
0.0010698291	0.0001099596	1.089	0.0001090107	1.553	0.0000009488	0.86295330%
0.0011403673	0.0001373443	0.001	0.0001373443	0.001	0.0000000000	0.00000000%

TABLE 8.10: Ex-post performance of two exemplary solutions.

	Solution 1 – Low return	Solution 2 – High return
Required daily return	0.0000118%	0.0011051%
Actual return stocks portfolio	-0.0009301%	0.0002369%
Actual return stocks and futures portfolio	0.0051194%	0.0003352%

which can be achieved through stock diversification alone.

The five instances, in which there was no difference between the asset constituents of the portfolios, were the ones with the highest required minimum returns. Generally, the gap decreased with increasing minimum returns. Two conclusions may be drawn. On the one hand, there is a threshold return, beyond which additional returns require a more significant increase in associated risk because this return is generally only achieved by fewer assets, reducing diversification benefits. For this set of data, it is found at 0.00114%. From this threshold on, the minimum required return could solely be achieved by certain assets, thus reducing the pool of potential assets and leading to portfolios of fewer included assets than the maximum number of five dictated by the cardinality constraint. This leads to portfolios composed of only stocks and thus also to a zero gap between the two portfolio types. On the other hand, it becomes obvious that the gap is much larger for low-return portfolios, from which one can conclude that risk-averse investors profit to a larger extent from futures diversification.

Ex-post stability analysis

Concerning the stability of the resulting portfolios, an ex-post application of the resulting portfolio weights has been conducted. Two of these are exemplarily presented below; the first corresponds to the lowest daily minimum return and the second corresponds to the highest minimum return level at which the portfolios still differed. A one-month ahead analysis is considered and the resulting returns of the portfolio are compared against the corresponding minimum return on a monthly basis. The ex-post analysis consists of the hypothetical investment into the best-found solution at the corresponding asset weights at the end of the sample period. Then, after a holding period of one month, the returns on the investment are calculated based on the actual price movements of the constituent assets.

Table 8.10 presents the metrics of the ex-post analysis for the first of two best-found portfolios presented in Figure 8.8 and 8.9, respectively. Solution 1 was found for an extremely risk-averse investor, while solution 2 represents a risk-loving investor's investment recommendation. The daily minimum return was a user-defined input for the proposed matheuristic. The actual return is presented below the minimum return for both the stocks and the combined stocks and futures portfolio. Exemplified by the two solutions in Table 8.10 the ex-post performance differs greatly depending on the required minimum return and the asset pool.

Figure 8.10 presents the risk-return characteristics of all ex-post portfolios for the ex-post period. At first sight, it can be constated that, solely taking into consideration the positive portion of

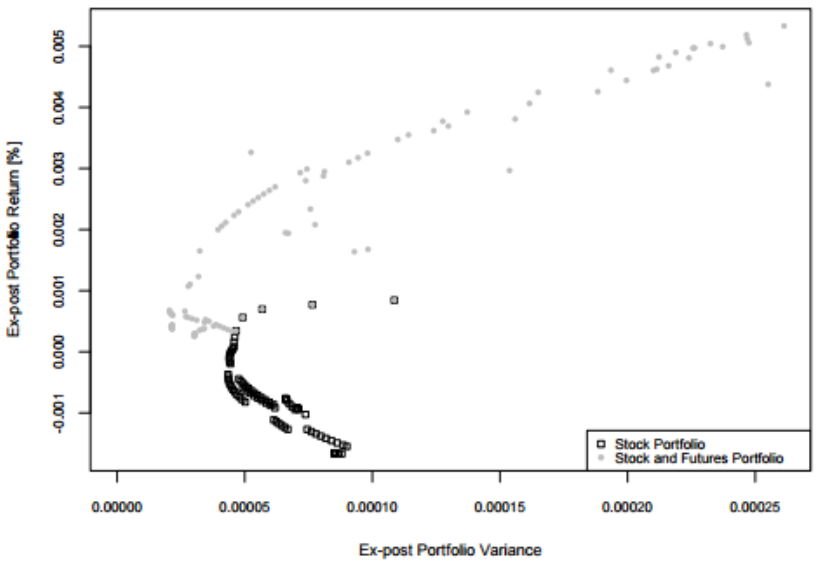


FIGURE 8.10: Frontiers of ex-ante optimal portfolios in ex-post analysis.

the return axis, the two curves resemble the shape of a Markowitz efficient frontier curve. It further becomes obvious that stock-and-futures portfolios overall achieved positive average ex-post daily returns, while a large portion of stock-alone portfolios presents the potential investor with negative returns. Because these negative returns are the result of downside risk exposure of the accompanying portfolios with high variance, it is intuitive that this part of the plot does not possess a positive slope. Concluding, it becomes evident that adding futures to the portfolios significantly improved the portfolio's behavior with respect to traditional financial theory in that increased returns can be achieved by assuming a more risk-exposed investment position. Moreover, the superiority of the stock and futures portfolios in ex-post performance is reinforced when considering both investment dimensions, risk and returns. Figure 8.10 shows that the ex-post portfolios of stocks and commodity futures can be a more effective vehicle of diversification than the portfolios of stocks only. Indeed, the ex-post portfolio variance is smaller for the former than for the latter, as shown by the minimum variance portfolio. Moreover, for a given value of the portfolio variance, average returns are larger for the portfolio combining both stocks and commodity futures. Furthermore, including commodity futures caters not only to risk-averse but also to risk-taking investors, since a broader range of values for both portfolio variance and return can be obtained.

In the following, it is analyzed whether the remaining instability of the portfolio weights was caused primarily by the risk or the return dimension of the portfolios. Figure 8.11 depicts the return dimension and presents a contrast of the minimum required return and the ex-post achieved one. If they were identical, the points should lie on the 45° line through the origin. However, due to the generally volatile nature of financial returns, this is not the case. The ex-post stock-alone portfolios significantly underperform the ex-ante portfolios and yield negative average daily returns for low and medium risk portfolios. Furthermore, while the ex-ante portfolio weights provide stable portfolio returns in that the minimum return is outperformed by the stock-and-futures portfolios for minimum returns on the lower spectrum of the analysis, the best-found solutions do not provide the minimum returns for extraordinarily high returns. While the latter result is a drawback to the investment, it is somewhat intuitive, as returns of such dimensions can only be achieved by assuming a significant level of risk. Moreover, the differences in the asset weights become almost negligible for minimum returns on the high end of the analysed spectrum. More strikingly, however, are the results for low return levels and thus risk-averse investors. Not only did the combined stock-and-futures portfolios generally outperform the traditional stock portfolio in the ex-post analysis, but also were they generally the only ones achieving an ex-post return of at least the originally required minimum return. This therefore reinforces the importance of futures diversification not solely from a risk perspective, but also from the perspective of stable short-term returns. However, the benefits of the portfolio optimization appear to be limited to lower risk portfolios and thus risk-averse investors from a return dimension perspective.

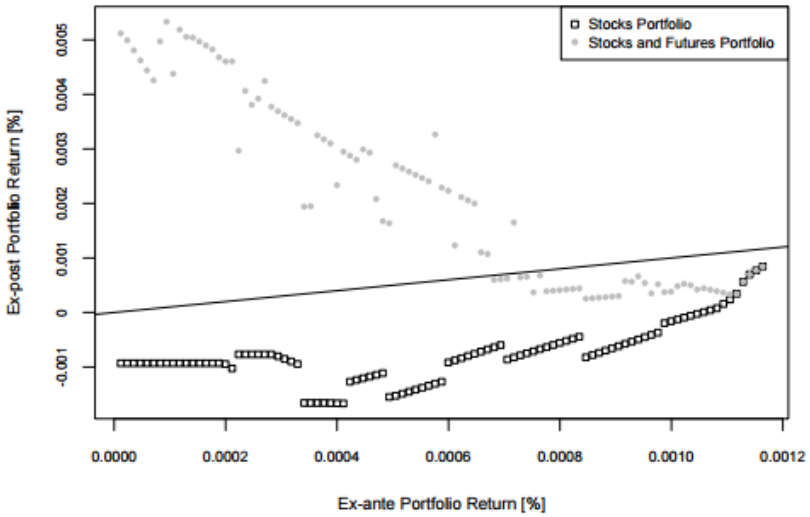


FIGURE 8.11: Comparison of ex-ante minimum required returns and ex-post actual returns.

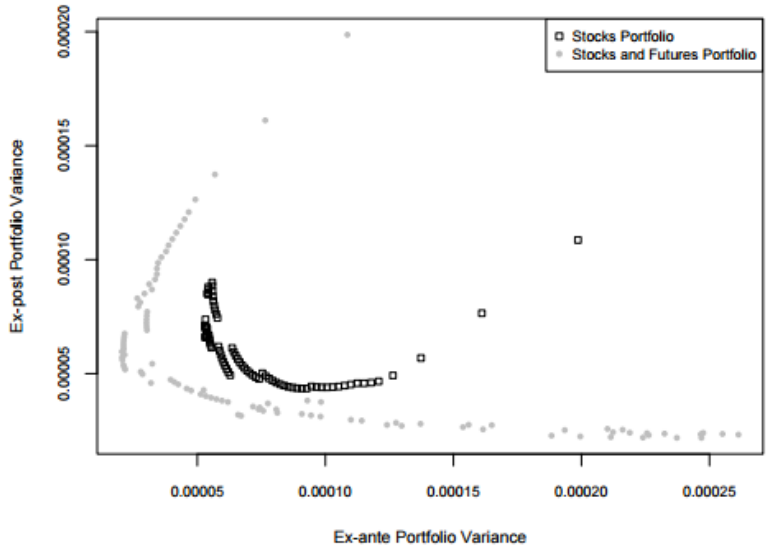


FIGURE 8.12: Comparison of ex-ante and ex-post portfolio risks.

Since the return dimension yields relatively stable performance for stock-and-futures portfolios, while this is not the case for stock-alone portfolios, it can be constated that this is a significant factor in distinguishing the different ex-post performances. Because, however, the ex-post combined portfolios also underperformed their ex-ante counterparts, the risk dimension is analyzed next. Overall, the non-stable risk variable of the portfolios seems to outweigh the volatility in returns in causing ex-post portfolio instability. Figure 8.12 presents a comparison of ex-ante risk and ex-post risk of the respective portfolios. If they were identical, the plotted points would lie on the 45° graph through the origin. As becomes obvious, low-levels of ex-ante risk are characterized by relatively non-stable ex-post risk levels and high variability. Contrary to that, high ex-ante risk level portfolios generally translate into lower-risk ex-post portfolios. It is to be noted further that the variability of the risk of stock-and-futures portfolios is higher than that of the stock portfolios. This variability then explains the level of ex-post instability of the combined portfolios.

Concluding, it can be stated that the best-found portfolios provide investors with stability in that the stock-and-futures portfolios outperform stock-alone portfolios in that they provide lower risk for a given minimum return level. However, the risk and return characteristics of the individual portfolios have shown to be instable over the observed period. While the return dimension mainly

causes the difference in performance between the ex-post stock-alone and stock-and-futures portfolios, the risk dimension explains the remaining instability between the ex-ante and ex-post performance of the combined portfolios.

8.6 Conclusions

Finance constitutes a highly dynamic and stochastic field playing an essential role in economy and social welfare. In this context, decision-makers frequently face diverse COPs such as the POP, in which an investor aims to select a few risky assets and decide the proportion of the budget to invest in each one in order to achieve a minimum return by minimizing a portfolio's risk measure. A richer version of the POP is defined by a number of additional constraints such as: cardinality, quantity and pre-selection constraints. This problem is usually tackled by using expected values for returns and covariances, which is an empirically unsupported assumption. A more realistic scenario is covered by the SPOP, where the aforementioned inputs are modelled as random variables. In addition to provide a review on metaheuristics applied to rich portfolio optimization and risk management, this chapter has presented solving methodologies based on metaheuristics and, for the SPOP, simulation. Aiming to facilitate the maximum diversification, a study is performed to quantify the benefit of introducing commodity futures to a portfolio of stocks.

The main conclusions are:

- The number of related publications has been increasing during the last decade, especially in the case of POPs. Population-based metaheuristics, and in particular GA and PSO, have been the predominant solving methodologies. Regarding single-solution metaheuristics, TS and SA have been extensively applied too. There is not a 'single winner' approach, meaning that different metaheuristic implementations have provided results of comparable quality to different problems.

- There is a clear trend in promoting the development of hybrid algorithms, either by combining different metaheuristics or by combining metaheuristics with statistical or machine learning techniques. However, there is a lack of works considering stochastic versions of the optimization problems.

- Most POPs include some kind of risk management and, in the other direction, most RMPs considering optimization issues can be modelled as enriched variants of POPs.

- The methodologies presented are able to solve real-sized instances in small amounts of time.

- Even in an environment with a relatively low level of variability, a stochasticity-aware approach may provide much better results than a metaheuristic approach generating solutions for the deterministic version of the POP.

- Futures contracts provide successful investment diversification. Particularly, risk-averse investors can drastically reduce their expected risk exposure by diversifying into stock-and-futures portfolios. Likewise, these portfolios of risk-averse investors yield more stable actual returns in the short term.

Chapter 9

Applications in computing

This chapter studies two important issues related to metaheuristics: the parameter fine-tuning and parallel computing. It presents a classification of works on parameter fine-tuning, and proposes a simple, general and automated methodology. However, a set of computational experiments on different COPs are carried out in order to analyze the effect of the number of agents and time on the performance of classical heuristics.
It is based on the following journal article: Calvet et al. (2016b).
This work has been presented at the following conferences: Calvet et al. (2015c) and Ruiz et al. (2015).

9.1 Introduction

Although the performance of metaheuristics is known to depend on its parameter values, the scientific community has not formally addressed the PSP until the end of the last century. According to Eiben et al. (1999), during the first decades of metaheuristics research, many scientists based their choices on tuning the parameters "by hand", i.e., experimenting with different values and selecting the ones that provide the best outputs, or "by analogy", applying settings that have been proven successful for similar problems. More recently, the need for a systematic approach towards setting of metaheuristic parameters has been increasingly outlined in the literature (Hooker, 1995; Johnson, 2002). Subsequently, researchers employ a scientific approach to tackle the PSP more frequently. It is important to highlight that the selection of a systematic methodology leads to a gain of efficiency, as in general, less time is required to fine-tune the parameters while the performance of the metaheuristic is the same if not improved. However, there is no methodology commonly accepted by the scientific community and there is also a lack of publications that compare, in an exhaustive and objective manner, the main approaches and the techniques used so far. Moreover, some of the proposed methodologies are not easily reproducible or are highly metaheuristic and problem dependent. These are some of the reasons why, in spite of the amount of parameter fine-tuning works, many practitioners go on tuning by hand or designing algorithms without parameters (or with a very low number of them), even in the case when more parameterized algorithms could lead to better performances. This chapter aims to contribute to the literature by proposing a general and automated statistical learning based procedure to tackle the PSP. It is accompanied by some methodological guidelines to validate the results. In order to test the methodology and illustrate its application, the approach is employed to fine-tune a hybrid algorithm implemented to solve the MDVRP.

The use of distributed and parallel computing systems (DPCS), which allows the aggregation of multiple autonomous computing resources interacting to achieve a common goal (Coulouris et al., 2005), may also have a significant effect on the performance of metaheuristics. This chapter also describes and tests an efficient, flexible, and browser-based framework to facilitate access to computational resources (Berry, 2009) and, ultimately, solve COPs in 'real time' (a few seconds). This framework enables the employment of new versions of web browsers (such as Google Chrome, Firefox, and Internet Explorer) as nodes in a cluster. The only required step is to visit a website. The embedded JavaScript code into this website enables the communication with the job dispatcher service. It may be considered a more scalable paradigm than traditional grid computing, since the connection of people is boosted by the fact that no third party software installation is required. Due to the relevance of COPs for SMEs and the some academic works proposing the implementation of DPCS for addressing them (Talbi, 2006; Talbi, 2009), the working and the

potential benefits of the proposed approach are illustrated here by solving the CVRP and the PFSP using different numbers of nodes running a simple metaheuristics with a given seed and a different limit of computational time.

9.2 Parameter fine-tuning

Ries et al. (2012) define the parameter setting problem (PSP) as the search for a set of parameter values θ^* in the parameter space Θ such that $\forall\theta \in \Theta : \theta^* \succeq \theta$ (where $\succeq$ denotes a relation of preference), for a given metaheuristic m in the metaheuristic space M, and a given instance x or group of them X in the instance space I. In practice, the amount of time available for experimenting T may be a restriction. In this case, the solution is approximate ($\hat{\theta}$). With regards to the difficulty of this problem, Montero et al. (2014) states that: (*i*) it is time consuming; (*ii*) the best set of parameter values depends on the problem at hand; and (*iii*) the parameters can be interrelated.

During the last decades, a large number of methodologies have been put forward to solve the PSP. These proposals can be classified in two groups (Birattari and Kacprzyk, 2009): parameter control strategies (PCS), and parameter tuning strategies (PTS). This classification is extended by instance-specific parameter tuning strategies (IPTS), which includes features of the aforementioned groups.

9.2.1 Literature review

This section provides a brief description of each approach and some of the most cited works. The interested reader is refered to more specific publications such as Eiben et al. (1999), De Jong (2007) and Battiti and Brunato (2010) for an expanded review of PCS, Birattari and Kacprzyk (2009) in the case of PTS, and Ries (2009) for IPTS.

Parameter control strategies

These methodologies aim for a dynamic fine-tuning of the parameters by controlling and adapting their values while solving a problem instance. They follow two basic steps: firstly, an initial set of parameter values is chosen; secondly, an adaptation mechanism is integrated which changes relevant parameter values. Most of these strategies apply adaptive parameter control, which means that their adaptation mechanism is based on the assessment of particular information that is stored during the iterative process of a metaheuristic. This information is usually related to the goodness of intermediate solutions. The main drawbacks of this approach are the potentially high computational effort required and the lack of acquired understanding about good parameter values each time an instance is solved.

Eiben et al. (1999) addressed the PSP in EAs. Three categories were defined to classify the PCS. The first one, deterministic parameter control, alters the value of a parameter by some deterministic rule, which is usually time based. The second category, adaptive parameter control, does employ feedback to determine the direction and/or magnitude of a parameter change. This is the most used kind of control. The third, self-adaptive parameter control (Smith, 2008), encodes the parameters to be adapted into the chromosomes of an EA. De Jong (2007) described the main motivations to use dynamic parameter setting strategies in EAs: first, as the running proceeds, information about the fitness landscape is generated, which may be used to improve the performance; also, changing the parameters is needed as an EA "evolves from a more diffuse global search process to a more focused converging local search process". Table 9.1 gathers a few representative works following this approach.

Parameter tuning strategies

This approach relies on the concept of robustness (Viana et al., 2005). A robust algorithm provides good results for a given set of instances of a problem using a fixed set of parameter values. The basic procedure involves finding a set of parameter values providing satisfactory results for a set of instances, usually using statistical and/or optimization techniques. Some authors analyse only a representative subset of instances and apply the set of parameter values found to solve all the instances. This approach also includes the case of solving one instance. Table 9.2 shows some works relying on this approach. Many authors focus on minimizing the number of runs, presenting

TABLE 9.1: Representative works employing PCS.

Work	Main techniques	Metaheuristic	Optimization problem
Battiti and Tecchiolli (1994) and Battiti and Brunato (2005)	Reactive scheme	TS	QAP and maximum clique problem
Zennaki and Ech-Cherif (2010)	SVMs	TS	TSP
Lessmann et al. (2011)	Regression models	PSO	Water supply network planning problem

simple models without interactions (e.g., Coy et al., 2001; Pongcharoen et al., 2007; Xu et al., 1998). DOE and regression analysis are the most employed techniques.

TABLE 9.2: Representative works implementing PTS.

Work	Main techniques	Metaheuristic	Optimization problem
Xu et al. (1998)	Tree growing and pruning method based on statistical tests	TS	Steiner Tree-Star Problem
Bartz-Beielstein et al. (2004)	DOE, classification and regression trees, and design and analysis of computer experiments	PSO and Nelder-Mead simplex algorithm	Elevator group controller problem
Birattari and Kacprzyk (2009) and Birattari et al. (2010)	Racing algorithm (Maron and Moore, 1993) and the Friedman's two-way analysis of variance by ranks (Conover, 1999)	ILS and ACO	QAP and TSP
Adenso-Diaz and Laguna (2006)	DOE and local search	Neighbourhood structure, TS, SA, TS, heuristic based on the SA and the TS, and TS	Steiner problem, part-machine grouping problem, part-machine grouping problem, single-machine scheduling, proportionate flowshops, and bandwidth packing
Pongcharoen et al. (2007)	DOE	GA	TSP
Ridge and Kudenko (2007)	DOE and desirability functions	ACO	TSP
Gunawan et al. (2013)	DOE, response surface methodology and ParamILS (Hutter et al., 2009)	SA	Industry spares inventory optimization problem

Instance-specific parameter tuning strategies

As in the case of PCS, IPTS aim for an instance-specific tailoring of the parameters. At the same time, these strategies use a fixed set of parameter values, as the PTS, avoiding the need of modifying the metaheuristic algorithm and reducing the potential computational effort required to adapt parameter values during the algorithmic run. In order to implement these strategies the relation between the parameter values and the performance of the metaheuristic has to be analysed, taking into account instance features. The next step consists in developing a mechanism able to use the features of a new instance to recommend a set of parameter values. The key element is the selection of instance features easy and fast to compute, and good at discriminating instances on the shape of their fitness landscapes, which analyse the relationship between the objective function values and the parameters. This learning may take a non-negligible amount of time, but it is assumed that this approach requires less computational time than the PCS approach does. Some contributions are included in Table 9.3. The number of works is low since it is relatively new.

Approaches comparison

All approaches have different advantages. The dynamic adaptation of the parameter values that characterizes PCS usually provides better results. However, the computational effort tends to be higher. On the other hand, the PTS approach is the easiest and fastest to use, once a set of parameter

TABLE 9.3: Representative works implementing IPTS.

Work	Main techniques	Metaheuristic	Optimization problem
Ries (2009)	DOE and fuzzy logic	Guided local search and GA	TSP
Pavón et al. (2009)	CBR and Bayesian networks	GA	Root identification problem
Dobslaw (2010)	DOE and NNs	PSO	TSP

values is selected. Although the code of the algorithm is not changed, finding an adequate set may be also time-consuming. The last group of strategies represents a compromise solution: it takes less computational time than the PCS approach, but requires implementing a learning mechanism, for which statistical learning skills are needed. Therefore, there is no approach that stands out from the others. Probably, the most adequate depends on the specific problem to tackle, the instances to solve, the available time and the skills of the researcher. Despite this fact, some general guidelines can be formulated. PTS can be considered as the best option when working with robust algorithms. Regarding IPTS, they are more complex than PTS but provide better results when the algorithm is not robust. In case of prioritizing the algorithm performance, PCS usually constitute the most recommendable approach.

9.2.2 Methodology

We propose a methodology that follows the PTS approach. As described before, this approach is not computationally intensive, and the inference from a representative sample of benchmark instances to the whole set usually provides good results, specifically if the analysed algorithm is robust. Another reason for focusing on PTS is that there is no methodology based on this approach and widely employed, but at the same time, there are plenty of techniques that can be used. Our methodology is based on clustering and DOE. The remainder of this section presents a statistical learning based methodology to obtain a list of sets of parameter values, and a more global procedure to validate and assess its goodness.

General methodology

It is assumed that the experimenter has described a problem and chosen the metaheuristic to tackle it.

- The first step involves choosing a subset of the instances. Their fitness landscapes will be analysed in order to obtain sets of parameter values that provide good results for them. The subset has to be representative as these sets of parameter values will be used to solve the whole set of instances. An approach to select a representative subset is, firstly, to determine the instance features that have a major influence on which set of parameter values is the most adequate, and then, choose the instances in such a way that the feature values of the subset are representative of those of the entire set of instances. For example, if there is a parameter for which its optimum value is known to depend on the instance size, a representative subset of the instances will present the same proportion of instances of a given size that the whole set does. A possible simplification for feature selection consists of choosing those that are commonly used to discriminate instances of a specific problem. For instance, Ries et al. (2012) studied the size, the distance metric, a ratio to describe the shape of the area within which a set of cities is distributed and a measure of clustering for the TSP.

  In contrast, a problem-independent approach is proposed here. Initially, for a given number of randomly generated sets of parameter values, each instance is solved several times using different seeds for the random number generator of the algorithm (or only once if the algorithm is deterministic). Alternatively, the sets could also be generated using more advanced statistical techniques such as DOE. We consider the median of the objective function values found with the same parameter values but different seeds. It is essential to remark the importance that a seed may have in the performance of an algorithm (Juan et al., 2015c; Czarn et al., 2004). Afterwards, feature scaling is applied to the values obtained for each instance.

Then, this data is used to cluster instances and select a representative one from each cluster. These instances form the subset to analyse.

For each instance of the subset, the steps ranging from the second to the fourth are implemented as follows.

- The second step requires selecting the range over which each parameter can be set. Some experience or knowledge about the problem and the metaheuristic may be highly valuable. The ranges should be large enough to cover at least one set of parameter values that can provide a sufficiently good solution with a high probability. On the other hand, a smaller range would allow the experimenter to describe more accurately, with the same resources, the relationship between the parameter values and the objective function value. If there is no a priori information about which are the best regions of the parameter space, a suitable procedure is to perform a rough and fast landscape analysis.
- The third step consists of designing an experiment. A central composite design is studied. Each parameter is considered a factor and the extreme values of its range define the levels. According to this design, the algorithm is executed also several times for each combination of factor values, each one with a different seed.
- In the fourth step, a procedure is developed to search the neighbourhood of the best set of parameter values found. Specifically, another central composite design centred on this set is applied.

Finally, the upshot is a list of recommended sets of parameter values, one per cluster; in particular, those that reported the best results on the last step. The procedure is shown in Figure 9.1. An extended proceeding (Figure 9.2) is described below in order to validate the list of sets of parameter values obtained and analyse the results provided by it.

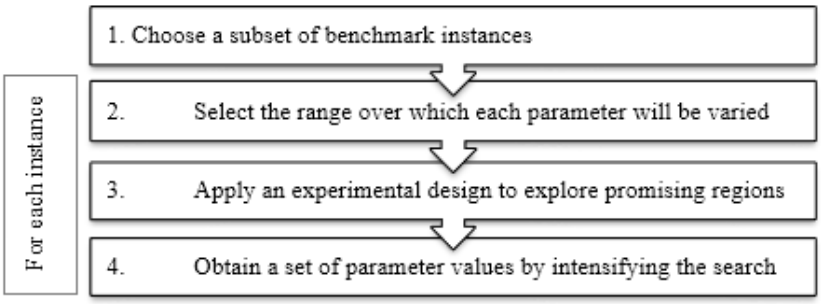


FIGURE 9.1: Outline of the procedure for parameter fine-tuning.

Before all else, a list of sets of parameter values, $\hat{\theta} = (\hat{\theta}_1, \hat{\theta}_2, \ldots, \hat{\theta}_K)$ where K is the number of clusters, is chosen as has been explained in the precedent subsection. Later on, each instance of the subset used to select $\hat{\theta}$ is solved with the corresponding set of $\hat{\theta}$, and with different sets, $\bar{\theta}_j$ ($j = 1, 2, \ldots, J$) (equally spaced, randomly selected or relatively close to the set of $\hat{\theta}$ according to some distance measure). To assess the performance of a set of $\hat{\theta}$ in a specific instance regarding the other sets, the associated solutions are compared. Given a decision level parameter r ($1 \leq r \leq J + 1$), if the rank of the objective function value provided by the proposed set is equal or lower than r, then it is considered a good set for that instance. Once all the instances of the subset are examined, it can be reckoned the proportion of them in which the corresponding set has been classified as good. $\hat{\theta}$ is validated by comparing this proportion with a predefined parameter p ($0 < p < 1$); if the proportion is higher, then the experimenter has enough evidence of the quality of $\hat{\theta}$ to go on to test it with other instances in the next step.

If $\hat{\theta}$ is not validated, the process has to be readjusted and restarted. This readjustment may be done in several ways, some options are: checking the robustness and the adequacy of the clustering, adapting the ranges, dedicating more resources to the search, etc. The best strategy is problem-dependent. As a consequence, the choice should rely on the opinion of the experimenter, who will have acquired valuable information from the outputs observed.

Once the list of sets of parameter values has been labelled as valid, it is applied for solving the other instances (each one with the set proposed for the representative instance of the cluster where

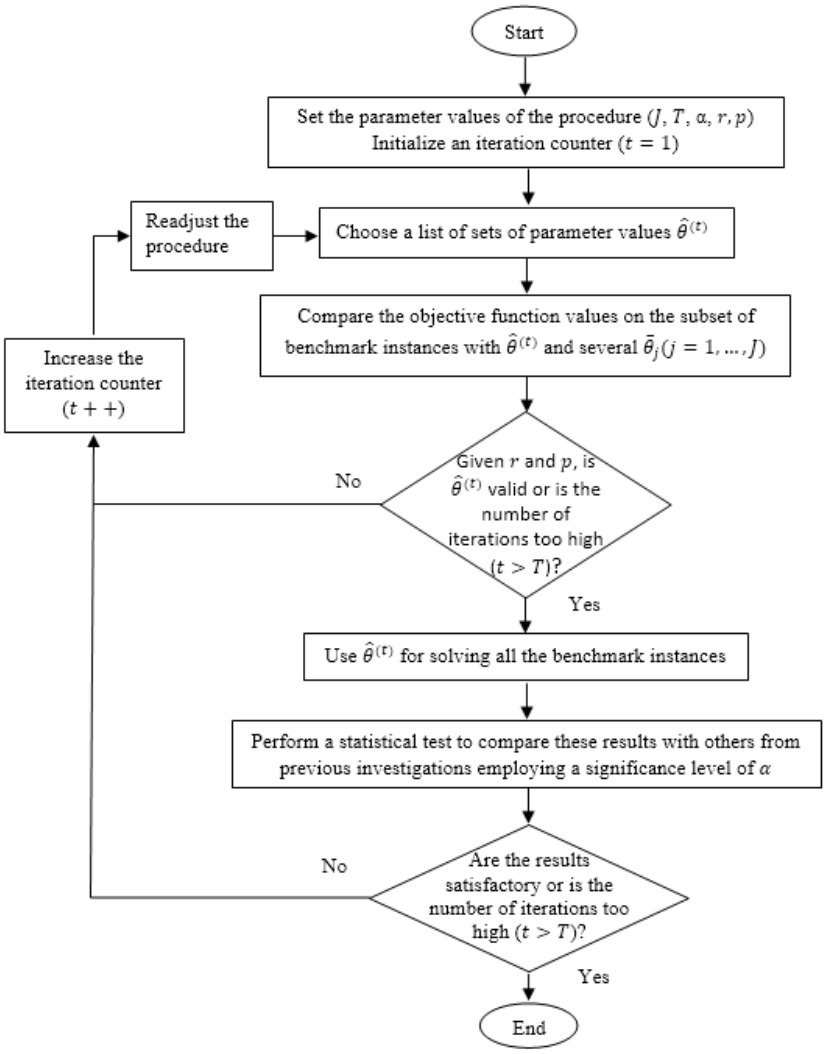


FIGURE 9.2: Flowchart representing the proposed methodology.

it has been assigned). To examine the effectiveness of the procedure, it is desirable to compare the solutions with others reported in the literature for the same instances, by performing the t-test for paired samples if data is normal, or the Wilcoxon signed rank test otherwise. If the means (or the mean ranks if data is not normal) do not differ significantly, it may be classified as a satisfactory outcome as it will mean that the proposed methodology, automated and general, has been proven to be competitive. If the results are unsatisfactory, the procedure should be modified and reinitiated.

It is useful to consider that, since the available resources are usually limited, the possible readjustments should be also limited (T represents this limit). Consequently, the process may end without a satisfactory list of sets of parameter values. In this case, the list which provides on average the best solutions will be accepted.

9.2.3 Computational experiments

Our methodology was implemented to fine-tune the parameters of the hybrid algorithm described in Juan et al. (2015c), which combines biased randomization and the ILS metaheuristic to address the MDVRP. This algorithm has three main parameters: bM, bR and $p*$, which take values between 0 and 1.

The first step is the selection of a representative subset of instances. Initially, 10 randomly generated sets of parameter values, 7 seeds and the 33 benchmark instances solved in the aforementioned paper were selected. Therefore, information from 2310 runs was stored. Data from different seeds was aggregated by computing the median; then feature scaling was applied. The instances that were considered easy-to-solve, those that presented no variation in the results, were

Table 9.4: Clustering of the benchmark instances.

Medoids	Clusters
p01	p01
p07	p04, p07, p11, p18, pr02, pr05, pr09
p09	p03, p09, pr04, pr10
p17	p17
p19	p19
p22	p22
p23	p20, p23
pr06	p05, p06, p08, p10, p15, pr01, pr03, pr06, pr07, pr08

separated. Afterwards, a clustering using the k-medoids algorithm (Theodoridis and Koutroumbas, 2009) was performed. The range of values considered for setting the value of k was 2-12. The final value was selected employing the average silhouette criteria (Rousseeuw, 1987). The composition of the clusters and the representative instances can be observed in Table 9.4.

Once the subset of instances was formed, the second step, setting the ranges of the parameters, was carried out. After a statistical analysis, it was concluded that just two parameters, bM and bR, did significantly affect the performance of the algorithm. Therefore, only those two parameters were studied. Five equally spaced values ranging from 0 to 1 were analysed for each parameter. Each instance was solved seven times (considering different seeds) for each possible combination of parameter values. The objective function values were aggregated as before. Then, the values for other possible combinations were estimated by linear interpolation.

The ranges were set to cover the smallest rectangular area of the parameter space that included the lowest objective function values. In particular, the values labelled as the lowest were those meeting the following condition:

$$\textit{Objective solution} \leq \textit{minimum value} + \beta \cdot (\textit{maximum value} - \textit{minimum value})$$

The value of β was set at a different value for each instance. More precisely, it was the minimum value that encompassed, at least, 5% of the search space. Figure 9.3 shows the contour plot and the area in which the search was intensified for each instance.

The next step was applying a design for each instance of the subset. It was performed to better analyse the relation between the metaheuristic performance and the parameter values. A face-centred central composite (FCC) design was selected, as in most of the cases the space parameter could not be expanded (since all parameters could only take values between 0 and 1). Figure 9.4 displays the scheme for instance p01. The objective function values for the same instance are represented in Figure 9.5.

Then, the neighbourhood of each set that provided the best solution for an instance was explored applying another FCC design, centred on that set and covering half of the area analysed with the previous design. The sets that finally presented the best performance were stored. They are outlined in Table 9.5. Random values were assigned to the instances that did not present variations in the results when changing the parameter values.

Table 9.5: Proposed list of sets of parameter values.

Medoids	Clusters	bM	bR
p01	p01	0.513	0.501
p07	p04, p07, p11, p18, pr02, pr05, pr09	0.001	0.372
p09	p03, p09, pr04, pr10	0.283	0.283
	p17	random	random
p19	p19	0.443	0.378
p22	p22	0.001	0.231
p23	p20, p23	0.449	0.250
pr06	p05, p06, p08, p10, p15, pr01, pr03, pr06, pr07, pr08	0.500	0.231
	p02, p12, p13, p14, p16, p21	random	random

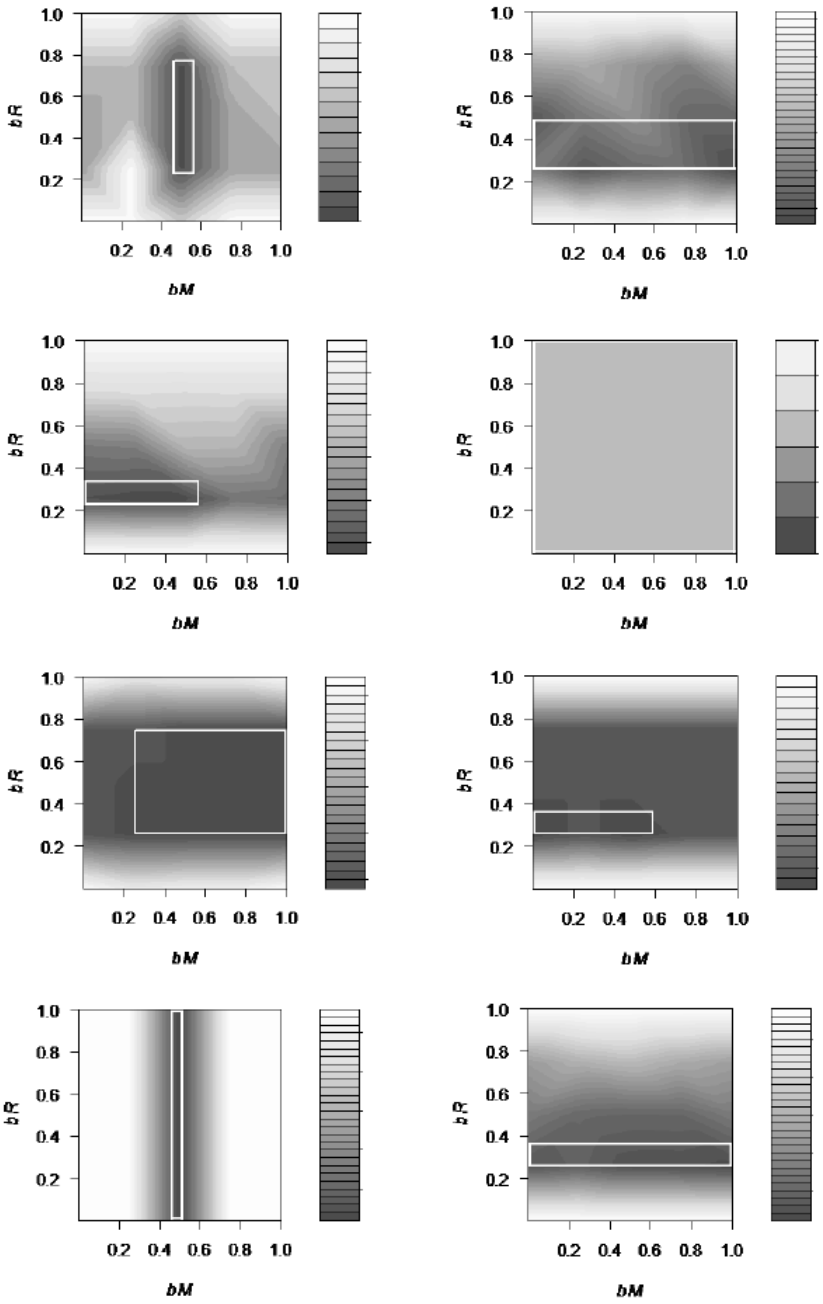


FIGURE 9.3: Contour plots of the medoids sorted from left to right, and top to bottom.

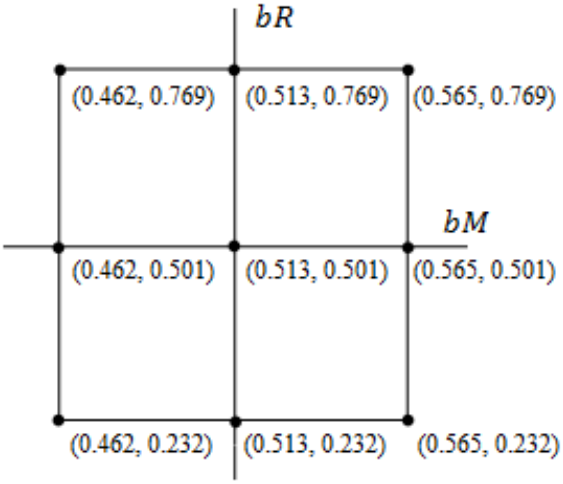


FIGURE 9.4: Scheme of the FCC design applied to the instance 'p01'.

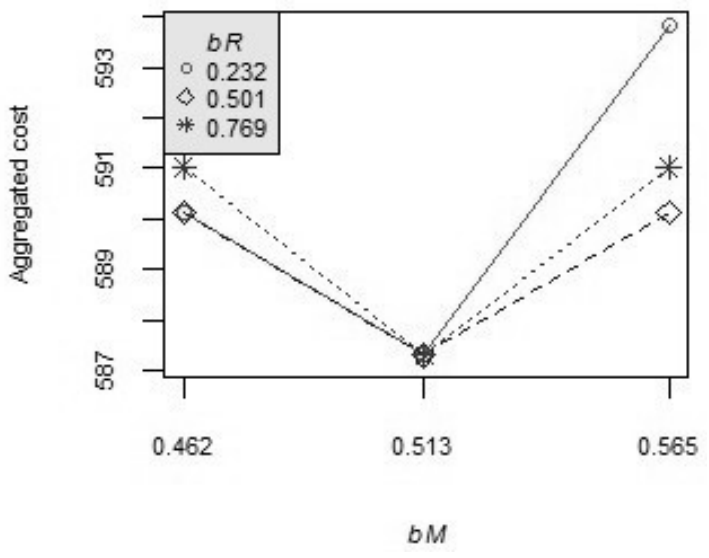


FIGURE 9.5: Solutions for the instance 'p01'.

9.2.4 Analysis of results

The following parameters were chosen to validate the list of sets: $J = 10$, $T = 3$, $\alpha = 0.05$, $r = 6$, $p = 0.7$. The number of sets randomly generated was fixed considering the trade-off between the reliability of our comparisons and the computational time required. The number of iterations was set considering only the time available. The significance level is the one most commonly used in the literature. The value of the fourth parameter is the mean rank that could be expected due to randomness with 11 solutions (1 set proposed and 10 randomly generated). The last parameter was calibrated to force the algorithm to provide good results at most of the instances. The algorithm was run 7 times with different seeds for each combination of parameter values, the medians and the minimum values were stored. The ranks of the results obtained are detailed in Table 9.6. Ties receive a rank equal to the average of the ranks they span, shown inside the parentheses.

TABLE 9.6: Ranks of the results provided by our list and by 10 random sets.

Medoids	Rank (medians)	Rank (minimum values)
p01	1	3.5 (1-6)
p07	5	3.5 (1-6)
p09	2	2
p17	2 (1-3)	1
p19	6.5 (2-11)	10.5 (10-11)
p22	11	11
p23	1.5 (1-2)	1
pr06	5	1.5 (1-2)
Valid instances	0.75	0.75

According to our methodology, the list of sets can be considered valid as it presents a rank equal to or below 6 in 75% of the analysed instances, both considering medians and minimum values. In order to test our results, the algorithm was executed with the parameter values suggested in Juan et al. (2015c). Both series of results are comparable as were obtained using the same computer and stopping criteria based on the number of iterations. Table 9.7 presents the parameter values used in the aforementioned paper. Instead of setting fixed values, the authors introduced randomness by employing uniform distributions. The lower and upper bounds were selected after some tests.

TABLE 9.7: Sets of parameter values for comparison.

bM	bR	p*
Uniform (0.5, 0.8)	Uniform (0.1, 0.2)	Uniform (0.1, 0.5)

Table 9.8 shows the results obtained solving all instances with the proposed list of sets (our results, OR), and with the set proposed in Juan et al. (2015c) (JR).

The comparison of the solutions shows that our procedure achieves better results in most of the instances. Table 9.9 presents the average and the standard deviation of the differences, and the p-values of the test to compare the mean ranks of the results. It is a non-parametric test as the null hypothesis of the Shapiro-Wilk test, a test of normality, was rejected in all cases. The means are negatives, indicating that our methodology provides better solutions. The p-values reveal that the differences of the mean ranks are not statistically significant. Even though, the magnitude of the mean difference can be considered relevant in the context of the MDVRP.

9.3 Parallel computing

Desktop computers have become affordable machines that most people use every day for both work and leisure. Despite their current capacity, numerous institutions and individuals require more computational resources to execute intensive problem-solving processes. In these cases, DPCS constitute a useful approach. Multi-processors and/or multi-computers paradigms may be employed. A multi-computer schema presents a set of physical machines linked via network connections. These machines can be coupled geographically or in a more distributed environment (as in cloud computing). The main parallel paradigm is message passing, in which tasks and

TABLE 9.8: Instances experimental results.

Inst.	OR medians (1)	OR, minimum values (2)	JR, medians (3)	JR, minimum values (4)	% Gap (1) - (3)	% Gap (2) - (4)
p01	585.000	576.866	593.829	576.866	-1.509	0.000
p02	480.261	476.660	480.261	476.660	0.000	0.000
p03	644.464	641.186	649.229	641.186	-0.739	0.000
p04	1022.085	1019.570	1024.473	1024.062	-0.234	-0.441
p05	760.341	756.281	764.325	754.882	-0.524	0.185
p06	882.827	879.072	880.418	879.763	0.273	-0.079
p07	899.709	897.974	906.395	897.974	-0.743	0.000
p08	4440.534	4434.552	4438.407	4426.747	0.048	0.176
p09	3920.743	3906.561	3923.248	3900.274	-0.064	0.161
p10	3706.763	3667.344	3705.012	3687.054	0.047	-0.537
p11	3598.972	3584.691	3592.891	3585.690	0.169	-0.028
p12	1318.955	1318.955	1318.955	1318.955	0.000	0.000
p13	1318.955	1318.955	1318.955	1318.955	0.000	0.000
p14	1360.115	1360.115	1360.115	1360.115	0.000	0.000
p15	2573.393	2556.846	2573.393	2557.528	0.000	-0.027
p16	2605.565	2585.373	2605.565	2600.099	0.000	-0.570
p17	2720.231	2714.663	2725.799	2725.799	-0.205	-0.410
p18	3831.996	3806.783	3835.388	3806.783	-0.089	0.000
p19	3883.686	3883.686	3883.686	3881.427	0.000	0.058
p20	4080.348	4074.779	4091.482	4091.482	-0.273	-0.410
p21	5706.530	5692.789	5701.902	5692.789	0.081	0.000
p22	5808.738	5806.370	5806.480	5786.288	0.039	0.346
p23	6134.441	6128.873	6145.576	6123.306	-0.182	0.091
pr01	861.319	861.318	861.319	861.318	0.000	0.000
pr02	1330.495	1310.679	1331.543	1314.364	-0.079	-0.281
pr03	1813.634	1813.634	1814.452	1813.634	-0.045	0.000
pr04	2084.843	2077.582	2089.785	2079.832	-0.237	-0.108
pr05	2379.075	2359.947	2379.797	2368.525	-0.030	-0.363
pr06	2709.792	2693.680	2713.593	2696.504	-0.140	-0.105
pr07	1109.235	1109.235	1109.235	1109.235	0.000	0.000
pr08	1680.896	1674.930	1678.872	1674.594	0.120	0.020
pr09	2148.216	2147.192	2153.317	2142.650	-0.237	0.212
pr10	3016.255	3008.129	3028.606	3014.874	-0.409	-0.224

TABLE 9.9: Means and standard deviations of the differences and statistical tests.

		Mean of the differences	Standard deviation of the differences	P-value of the comparison of mean ranks
All instances	Medians	-0.149	0.330	0.954
	Minimum values	-0.070	0.219	0.980
All instances except the studied subset and those not analysed	Medians	-0.117	0.247	0.942
	Minimum values	-0.100	0.217	0.942

processes of different machines interchange data packets by sending and receiving messages to communicate.

SMEs are responsible for a significant part of the wealth generated in all developed economies. Often, they do neither possess advanced technical knowledge nor modern computational resources. However, a number of them could benefit from having more resources, for example to speed up intensive-computation processes or to obtain a higher performance. In order to access them, DPCS offer two alternatives: *(i)* to pay for using resources from an external provider; and *(ii)* to employ underutilized computer resources owned by the SME. This idea of aggregating idle or unused resources characterizes also volunteer computing systems. The main difference between both paradigms is that while the latter is usually associated to dynamic (any user can freely enter and leave) and heterogeneous environments, an SME knows the characteristics and the availability of its machines. Obviously, their scalability is also more limited. The alternative of using SME's underutilized resources presents several advantages. Firstly, SMEs do not have to send private information to servers of an external enterprise. Secondly, it is a cheaper solution since the SME does already have the resources. Finally, the energy consumption is reduced by seizing these resources, which could be still consuming otherwise (Cabrera, 2014). These desktop grids systems may be formed by personal computers with more computing capabilities than the required (standard computers in which employees mainly use word processors and spreadsheets, for instance)

or that are not used during some specific days or hours. Moreover, resources from several SMEs may be gathered to build a larger computing system. They can rely on a directory-of-resources service that keeps updated information of available computing resources. Once a user requires executing a process, he sends a query to this directory to select the resources and organize the tasks to perform. Once these tasks have been completed, the result is sent back to the user.

9.3.1 Methodology

The platform designed aims at facilitating the aggregation of a high number of computational resources by seizing underutilized or idle resources. It is based on software already installed in most computers, web browsers. Using a modern version of some of the commonest (Google Chrome, Firefox, or Internet Explorer), it may integrate a computer into the computing network. The only action required is to visit a website with an embedded JavaScript code that enables the communication in real-time with a job dispatcher service. Each one of the jobs includes a piece of data and the computing task to perform. Additional steps such as downloading, installing, or setting up additional software are not required, which makes this option a very attractive one for most SMEs. Because the ease to add new resources, this approach can be considered highly scalable. Thus, the described platform constitutes a flexible, simple, and scalable approach with multiple applications in SMEs.

The platform architecture is the typical of a master-slave cluster. The system has been designed to free the master from computationally expensive tasks. For the experiments described later, a single master has been sufficient to handle all the workload. In a production environment, the system could easily scale to thousands of slaves or even further when considering other architectures like a multi-master environment, etc. In our case, the master was placed on a dedicated server located on a cloud provider (Softlayer). The slaves were located over 2 different locations: The UOC's Lab and the Incubio's offices. The execution process goes as follows. First, the end-user submits the task to be executed to the master. This task consists mainly in a set of Map and Reduce functions written in JavaScript, as well as their input dataset. The master is responsible of creating a list of jobs. Each job is composed of a chunk from the dataset and the source of code that has to be executed over each piece of data. The master delivers and ensures that jobs are evenly distributed. After each job is completed, the master receives the results and stores them in a file or a database depending on the execution flow given by the user. The master keeps track of the jobs that have been assigned and processed. Different measures handle unfinished jobs, errors or exceptions that could appear unexpectedly by either rescheduling the jobs or stopping the execution and reporting the error.

Most approximate methods for solving COPs are probabilistic, which means that their solution depends on the seed used for a pseudo-random number generator. It has been proved that the execution time that an algorithm needs to report high-quality solutions can be reduced depending on this seed (Juan et al., 2014d). According to Talbi (2006) and Talbi (2009), DPCS are commonly employed to solve COPs. The typical approach in the related literature applies a master-slave scheme, in which a master or coordinator processor sends tasks to a set of slave processors in order to execute an intensive-computing process. Each slave is responsible for solving the same problem instance considering a different scenario, each one formed by a set of parameters and/or a seed. Once a slave has completed its task, it sends the solution to the master that stores it. In the simplest version, there is no communication between slaves. Following this approach, multiple instances of the algorithm are executed at the same time, each with a different seed. As shown in Figure 9.6, each of these instances can be considered a cloned agent that is searching the solution space.

9.3.2 Computational experiments

In order to illustrate the benefits of the presented approach, two relevant COPs have been addressed: the CVRP and the PFSP. The randomized version of the CWS heuristic (Juan et al., 2014d) has been chosen for the CVRP. The Kelly instances are used to test our approach (Golden et al., 2008). The ILS-ESP algorithm (Juan et al., 2014a) has been employed to address the PFSP. It relies on a biased-randomized version of the NEH heuristic. The Taillard's benchmark instances (Taillard, 1993) are employed. They are grouped in 12 sets of 10, which are characterized by the following pairs of numbers of jobs and machines: 20x5, 20x10, 20x20, 50x5, 50x10, 50x20,

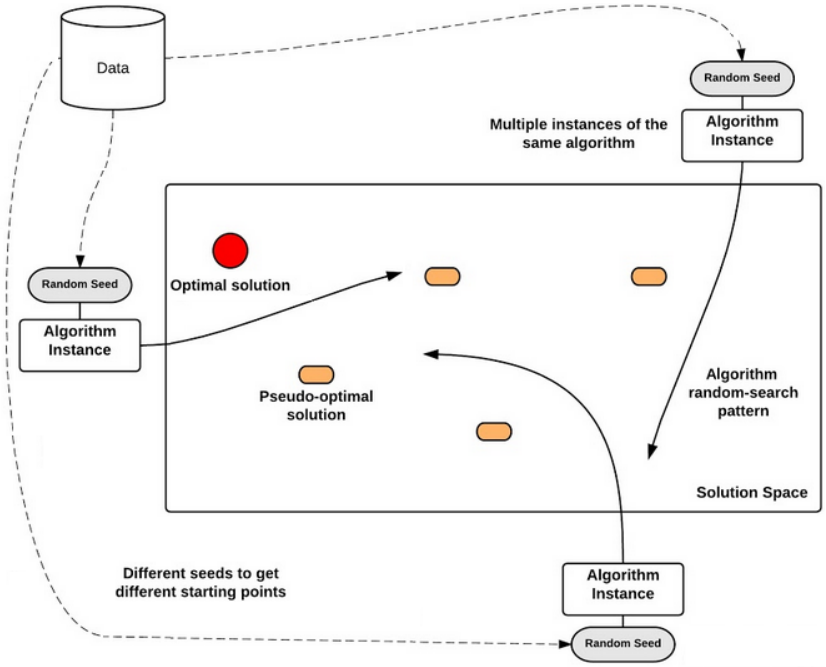


FIGURE 9.6: A multi-agent approach for solving COPs.

100x5, 100x10, 100x20, 200x10, 200x20, and 500x20. The instance resolution has been performed considering a specific combination of the parameters 'limit of time' (1, 5, 10, 15, 20, and 30 seconds) and 'number of agents' running in parallel (1, 4, 8, 16, 32, and 64).

All the experiments have been carried out using 64 slaves and 1 master. The master specifications are 3.5GHz Intel Xeon-IvyBridge with 8GB of RAM. The slaves are a heterogeneous set of desktop computers not having more than 8GB of RAM and up to 8 cores each. The machines were connected to the parallel computing environment using one of the following browsers: Microsoft Internet Explorer, Google Chrome or Mozilla Firefox, all of them with JavaScript enabled. The slaves were connected over a usual shared internet connection. For this reason, latencies or high speed connections were considered negligible.

9.3.3 Analysis of results

Figure 9.7 shows the results obtained after running the algorithm for solving the Kelly instances during 20 seconds of clock time per instance. Considering all instances, the first boxplot shows the gaps between the BKS and the solution generated by the CWS heuristic. The remaining boxplots show the gaps between the BKS and different executions of the randomized algorithm, each one using a different number of agents running in parallel. The number of agents tested were: 64, 128, and 256. It should be noticed that, for the 20 seconds considered, the distributed approach allows to reduce the gap down to almost 5% even for a reasonably low number of agents.

Regarding the PFSP instances, Table 9.10 summarizes the results of our computational experiments using a maximum time of 5 seconds. Each row refers to a different set of instances. Each column shows the gap between the BKS and our solution for different numbers of agents (1, 4, 8, 16, 32, and 64). Notice that the gaps shrink as the number of agents working in parallel is increased.

Figure 9.8 summarizes similar results for different values of the maximum clock time. It can be observed that, as time increases or as the number of agents increases, the average gap (for the entire set of benchmark instances) decreases. A detailed case is illustrated in Figure 9.9, which displays the scatterplot of costs versus limit of time and number of agents for a given instance.

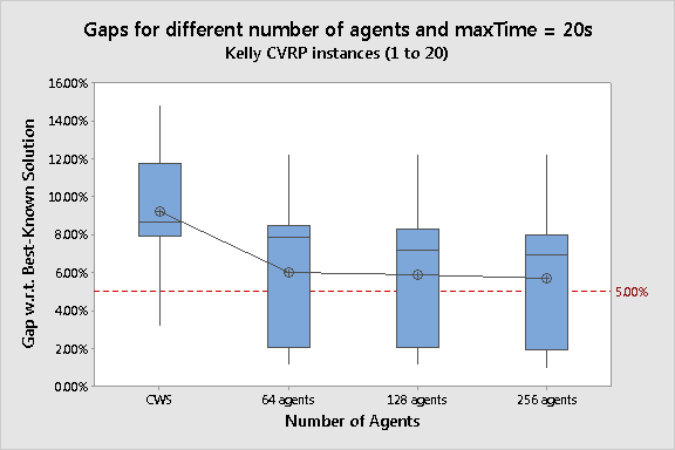


FIGURE 9.7: Results for the CVRP using the Kelly instances.

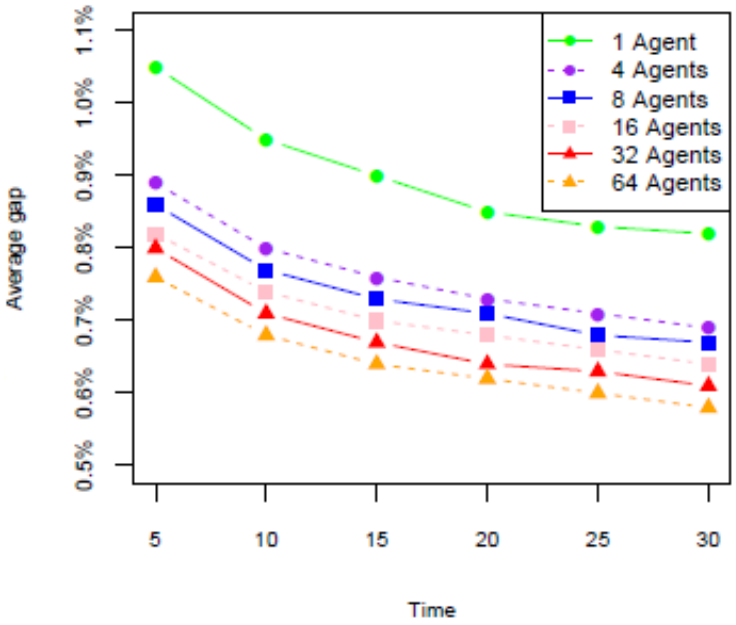


FIGURE 9.8: Average gaps for different numbers of agents and limits of time.

TABLE 9.10: Results for the PFSP considering the Taillard instances. Gaps for different number of agents and a maximum time of 5 seconds.

Taillard set	BKS - 1A	BKS - 4A	BKS - 8A	BKS -16A	BKS -32A	BKS - 64A
20x5	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
20x10	0.08%	0.08%	0.04%	0.00%	0.00%	0.00%
20x20	0.06%	0.02%	0.01%	0.00%	0.00%	0.00%
50x5	0.01%	0.01%	0.01%	0.01%	0.00%	0.00%
50x10	0.84%	0.74%	0.72%	0.65%	0.61%	0.54%
50x20	3.21%	2.85%	2.76%	2.68%	2.65%	2.58%
100x5	0.05%	0.02%	0.00%	0.00%	0.00%	0.00%
100x10	0.53%	0.33%	0.25%	0.22%	0.21%	0.18%
100x20	3.14%	2.67%	2.66%	2.63%	2.56%	2.46%
200x10	0.40%	0.26%	0.24%	0.23%	0.20%	0.14%
200x20	2.36%	2.20%	2.17%	2.15%	2.06%	2.00%
500x20	1.88%	1.53%	1.42%	1.32%	1.32%	1.26%
Averages	1.05%	0.89%	0.86%	0.82%	0.80%	0.76%

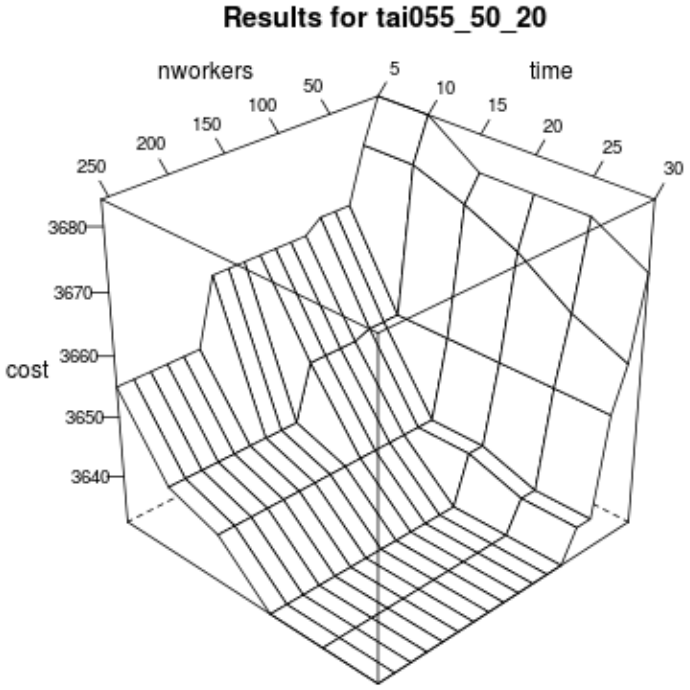


FIGURE 9.9: Objective solutions for different numbers of agents and limits of time.

9.4 Conclusions

The performance of metaheuristics is significantly affected by the parameter fine-tuning and the number of agents employed. These issues have not attracted as much attention as new metaheuristics and applications did. However, this trend is changing. In these lines, this chapter has presented two contributions. First, an overview on the PSP, a classification, a methodology and a description of a case study have been presented. Afterwards, an analysis of the computational time and the number of agents (each using a different seed for the random number generator) on the performance of two simple algorithms has been described. The conclusions drawn are:

- The parameter fine-tuning of a metaheuristic may be a time-consuming and complex problem, but may have a high effect on the quality of the solutions found.
- The literature on the PSP is diverse. Works can be grouped by whether the set of parameter values chosen is instance-specific and/or whether it evolves during the execution of an algorithm.

- A number of statistical learning techniques for solving the PSP have been proposed in the literature, but most works focus on DOE and/or lineal regression models. Thus, many options remain to be tested.
- SMEs may significantly benefit from DPCS by obtaining a higher number of computational resources seizing the underutilized resources.
- Computational time tends to have a small marginal effect on the performance of metaheuristic when it is set to more than a few seconds.
- The number of runs with a different seed executing a given metaheuristic may have an important effect on its performance, but the marginal improvement decreases as the number of runs increases.

Part III

CONCLUSIONS, FUTURE RESEARCH AND CONTRIBUTIONS

Chapter 10

Conclusions and future work

10.1 Final conclusions

This thesis has explored the combination of statistical learning and simulation with metaheuristics for solving COPs. It includes both methodological contributions and applications in a wide range of relevant and challenging fields.

First, an extensive review on works using statistical learning and metaheuristics as a solving approach has been presented. While there is a high number of works, they are extremely different. Two groups are created: metaheuristics for improving statistical learning, and statistical learning for enhancing metaheuristics. Works in the first group can be classified according to the purpose of the statistical learning technique: classification, regression, clustering, and rule mining. On the other hand, the second group is splitted into two smaller groups: specifically-located hybridizations (including parameter fine-tuning, initialization, evaluation, population management, operators, and local search), and global hybridizations (reduction of search space, algorithm selection, hyperheuristics, cooperative strategies, and new types of metaheuristics).

Then, a novel hybrid methodology integrating statistical learning in metaheuristic frameworks has been proposed. It has been designed to address combinatorial optimization problems with dynamic inputs, which depend on the structure of the solution. A number of potential applications in popular fields have been identified, and an illustrative experiment has been carried out.

Applications to transportation constitute the main topic in applications. The multi-depot vehicle routing problem has been introduced, and three richer novel extensions have been addressed. First, uncertainty regarding demands has been considered. This issue may increase the total expected costs if no measures are undertaken to reduce the probability of route failures. A simheuristic approach considering safety stocks has been designed and tested. Afterwards, a version considering the maximization of benefits, heterogeneous depots and customers' preferences has been studied. In order to solve it, a methodology combining predictive models and a metaheuristic has been put forward. The third extension considers the introduction of sustainability indicators in the objective function and presents an analysis relying on visualization techniques to study the relationship between the different indicators. The aim is to take into account the negative impacts of transport activities. Later, the waste collection problem has been addressed, presenting methodologies for both the deterministic and stochastic versions. Finally, the heterogeneous site-dependent asymmetric vehicle routing problem with stochastic demands has been tackled and a simheuristic methodology based on a successive approximations method has been applied. The potential applications of these problems in real-life has been described.

Regarding production, the distributed permutation flow-shop scheduling problem with stochastic times has been introduced. It describes the realistic scenario where there is a product composed of several intermediate products that have to be assembled at a given moment. These intermediate products are processed in independent distributed manufacturing factories, and each sub-problem is modeled as a permutation flow-shop scheduling problem. An approach relying on a simheuristic algorithm based on the iterated local search metaheuristic has been presented and tested.

Metaheuristics are becoming popular in finance. A survey on metaheuristics in portfolio optimization and risk management has been presented. Afterwards, the deterministic and stochastic versions of the portfolio optimization problem have been adressed. This problem is a strategy of selection of financial assets and determination of the optimal weights allocated to those assets that results in a desired portfolio return and associated minimum level of risk. The stochastic version deals with returns and covariances modelled as random variables.

Finally, two issues related to computing have been studied: the parameter fine-tuning, and the effect of the number of seeds, and the maximum computing time on the performance of metaheuristics. While a methodology is presented and applied for the first issue, an exhaustive set of computational experiments have been carried out to gain insights into the second.

10.2 Directions for future work

Numerous lines of future research stem from this thesis. They are summarized in the following proposals:

- Extend the methodology of learnheuristics to address stochastic and/or multi-objective optimization problems, and develop an online version, in which information regarding new inputs can be used to improve the predictive model, and a blended version, in which predictions from several models are averaged, not necessarily giving the same weight to each of them.
- Design and test more approaches relying on learnheuristics for problems in dynamic fields such as telecommunications, volunteer computing or finance.
- Several rich vehicles routing problems have been addressed. Many realistic characteristics may be added, which could increase the complexity of the problems. It would be interesting to study the efficiency of repairing procedures for a given solution when unexpected events take place. In addition, large supply chains with flexible structures with more agents than clients and depots could be included.
- Production systems have dramatically changed during the last decades, and some gaps remain in the literature. The most natural extension of the problem analyzed is to study the effects of dependent processing times.
- There are a high number of non-trivial optimization problems in finance. The uncertainty/risk of this field calls for the combination of optimization techniques and predictive models, and/or online optimization. Moreover, it would be interesting to analyze the impact of the width of the sample period and associated bear and bull market activity periods for the problems studied.
- In the computing arena, the calibration of parameters is still an open problem, since there is no single methodology accepted by the scientific community. Even more, there is no general agreement about the best techniques to use. However, journals devoted to applications of metaheuristics are becoming more demanding regarding this issue. In addition, the trade-off between number of computational experiments, maximum computing time, and performance requires more attention. Similarly, the use of parallel and distributed paradigms are just emerging in the field of metaheuristics. In an increasingly complex world where real-time solving approaches are required and statistical learning techniques may help to develop more inteligent/reactive approaches, these paradigms will play a relevant role.

Chapter 11

List of publications and presentations

This chapter lists the publications and presentations related to this thesis. It includes the accepted or in process of reviewing journal papers, and works in conferences and seminars developed in the last three years.

The relation between the publications and presentations, and the contributions described in chapter 1 is shown below.

- C1 (approach combining metaheuristics and statistical learning, general classification and review of works): J1, J2, S1, and S2.
- C2 (applications in routing): J4, J7, J12, J15, C3, C6, C8, and C10.
- C3 (applications in production): J10, C1, and C5.
- C4 (applications in finance): J5, J8, J9, J11, J14, J17, C2, and C9.
- C5 (applications in computing): J3, C4, and C7.
- Other works related to the methodologies employed: J6, J13, and J16.

11.1 Journal papers

First, the following articles have been submitted to ISI JCR and Elsevier-Scopus journals:

Indexed in ISI JCR

J1. **Calvet, L.**, J. De Armas, D. Masip, and A. A. Juan (2017). "Learnheuristics: Hybridizing metaheuristics with machine learning for optimization problems with solution-dependent inputs". In: *Open Mathematics* (indexed in ISI SCI, 2015 IF = 0.512, Q3; 2015 SJR = 0.521, Q2).

J2. **Calvet, L.**, A. Ferrer, I. Gomes, A. A. Juan, and D. Masip (2016). "Combining statistical learning with metaheuristics for the multi-depot vehicle routing problem with market segmentation". In: *Computers and Industrial Engineering* 94, pp. 93–104 (indexed in ISI SCI, 2015 IF = 2.086, Q1; 2015 SJR = 1.63, Q1).

J3. **Calvet, L.**, A. A. Juan, C. Serrat, and J. Ries (2016). "A statistical learning based approach for parameter fine-tuning of metaheuristics". In: *Statistics and Operations Research Transactions* 40.1, pp. 201–224 (indexed in ISI SCI, 2014 IF = 1.333, Q2; 2014 SJR = 0.324, Q3).

J4. Gruler, A., C. Quintero, **L. Calvet**, and A. A. Juan (2016). "Waste collection under uncertainty: A simheuristic based on variable neighborhood search". In: *European Journal of Industrial Engineering* (indexed in ISI SCI, 2014 IF = 0.736, Q3; 2014 SJR = 0.898, Q1).

Under review

J5. **Calvet, L.**, R. Kizys, A. A. Juan, and J. Doering (submitted). "A VNS-based simheuristic methodology for the stochastic portfolio optimization problem". In: *Journal of the Operational Research Society*.

J6. **Calvet, L.**, M. Lopeman, J. De Armas, G. Franco, and A. A. Juan (submitted). "Statistical and machine learning approaches for the minimization of trigger errors in earthquake catastrophe bonds". In: *Statistics and Operations Research Transactions*.

J7. **Calvet, L.**, D. Wang, and A. A. Juan (submitted). "A simheuristic algorithm for the stochastic multi-depot vehicle routing problem". In: *International Transactions in Operational Research*.

J8. Doering, J., **L. Calvet**, A. Fito, R. Kizys, and A. A. Juan (submitted). "Metaheuristics for realistic portfolio optimization and risk management: Current state and future trends". In: *Annals of Operations Research*.

J9. Doering, J., **L. Calvet**, R. Kizys, A. Fito, and A. A. Juan (submitted). "Rich portfolio optimization with stocks and individual commodity futures contracts". In: *OR Spectrum*.

J10. Hatami, S., **L. Calvet**, V. Fernandez-Viagas, J. Framinan, and A. A. Juan (submitted). "Combining simulation with metaheuristics in distributed scheduling problems with stochastic processing times". In: *International Transactions in Operational Research*.

J11. Kizys, R., A. A. Juan, B. Sawik, and **L. Calvet** (submitted). "ARPO: An iterated local search algorithm for portfolio optimization under realistic constraints". In: *Quantitative Finance*.

J12. Reyes, L., **L. Calvet**, C. Talens, A. A. Juan, and J. Faulin (submitted). "Sustainable Urban Freight Transport: a multi-depot vehicle routing problem considering different cost dimensions". In: *Journal of Heuristics*.

Indexed in Elsevier-Scopus

J13. **Calvet, L.** and A. A. Juan (2015). "Educational data mining and e-learning analytics: An overview of goals, quantitative methods, and time-line evolution". In: *International Journal of Educational Technology in Higher Education* 12.3 (indexed in ISI ESCI, 2014 SJR = 0.215, Q3).

J14. **Calvet, L.**, A. A. Juan, R. Kizys, and J. De Armas (2016). "A SimILS-based methodology for a portfolio optimization problem with stochastic returns". In: *Springer Lecture Notes in Business Information Processing* 254, pp. 3–11 (indexed in ISI Web of Science and Scopus, 2014 SJR = 0.244, Q3).

J15. **Calvet, L.**, A. Pages, O. Travesset, and A. A. Juan (2016). "A simheuristic for the heterogeneous site-dependent asymmetric VRP with stochastic demands". In: *Springer Lecture Notes in Computer Science / LNAI* 9868, pp. 408-417 (indexed in ISI Web of Science and Scopus, 2014 SJR = 0.339, Q2).

J16. De Armas, J., **L. Calvet**, G. Franco, M. Lopeman, and A. A. Juan (2016). "Minimizing trigger error in parametric earthquake catastrophe bonds via statistical approaches". In: *Springer Lecture Notes in Business Information Processing* 254, pp. 167-175 (indexed in ISI Web of Science and Scopus, 2014 SJR = 0.244, Q3).

J17. Doering, J., A. A. Juan, R. Kizys, A. Fito, and **L. Calvet** (2016). "Solving realistic portfolio optimization problems via metaheuristics: A survey and an example". In: *Springer Lecture Notes in Business Information Processing* 254, pp. 22–30 (indexed in ISI Web of Science and Scopus, 2014 SJR = 0.244, Q3).

11.2 Conferences and seminars

Some works have been presented in conferences or seminars:

Indexed in ISI-WOS or Elsevier-Scopus

Co1. **Calvet, L.**, V. Fernandez-Viagas, J. Framinan, and A. A. Juan (2016). "Combining simulation with metaheuristics in distributed scheduling problems with stochastic processing times". In: *Proceedings of the 2016 Winter Simulation Conference*. Washington D. C., USA, pp. 2347–2357.

Peer-review conferences

Co2. **Calvet, L.**, J. Doering, R. Kizys, A. A. Juan, and A. Fito (2016). "The stochastic portfolio optimization problem: A formulation and a hybrid methodology". In: *OR58 Annual Conference*. Portsmouth, UK, pp. 127–128.

Co3. **Calvet, L.**, A. A. Juan, and N. Schefers (2015). "Solving the multi-depot vehicle routing problem considering uncertainty and risk factors". In: *Proceedings of the ICRA6 / Risk 2015 Int. Conference*. Barcelona, Spain, pp. 187-194.

Co4. **Calvet, L.**, A. A. Juan, and C. Serrat (2015). "Técnicas estadísticas aplicadas a la calibración de parámetros de metaheurísticas". In: *Proceedings of the X Congreso Español de Metaheurísticas, Algoritmos Evolutivos y Bioinspirados*. Mérida, Spain, pp. 409–416.

Co5. **Calvet, L.**, M. Mateo, A. A. Juan, and C. Laroque (2016). "Optimizing starting times in parallel multiple production lines with stochastic processing times and a shared deadline". In: *Proceedings of the 15th International Conference on Project Management and Scheduling*. Valencia, Spain.

Co6. Juan, A. A., J. Faulin, and **L. Calvet** (2015). "Supporting real-time decision-making in logistics and transportation by combining simulation with heuristics". In: *Proceedings of the 12th Int. Multidisciplinary Modeling & Simulation Conf.* Bergeggi, Italy, pp. 35–38.

Co7. Ruiz, X., **L. Calvet**, J. Ferrarons, and A. A. Juan (2015). "SmartMonkey: a web browser tool for solving combinatorial optimization problems in real time". In: *Proceedings of the 2015 Int. Conf. of the Forum for Interdisciplinary Mathematics*. Barcelona, Spain.

Other international conferences

Co8. **Calvet, L.**, A. Ferrer, A. A. Juan, and I. Gomes (2015). "Market segmentation issues in the multi-depot vehicle routing problem". In: *EURO 2015*. Glasgow, UK.

Co9. Doering, J., **L. Calvet**, R. Kizys, A. Fito, and A. A. Juan (2016). "Rich portfolio optimization with stocks and individual commodity futures contracts". In: *Portsmouth-Fordham Conference on Banking & Finance*. Portsmouth, UK.

Co10. Juan, A. A., J. Faulin, **L. Calvet**, A. Pages, and C. Quintero (2015). "Applications of simheuristics in transportation and logistics". In: *EURO 2015*. Glasgow, UK.

Seminars

S1. **Calvet, L.** (2015). "Hybridizing machine learning and metaheuristics". YouBRA Workshop. Barcelona, Spain.

S2. **Calvet, L.** (2015). "Neural networks for routing problems: review and challenges". Green COOP-CYTED Workshop. Madrid, Spain.

Bibliography

Adenso-Diaz, B. and M. Laguna (2006). "Fine-tuning of algorithms using fractional experimental designs and local search". In: *Operations Research* 54.1, pp. 99–114.

Adibi, M. A. and J. Shahrabi (2013). "A clustering-based modified variable neighborhood search algorithm for a dynamic job shop scheduling problem". In: *The International Journal of Advanced Manufacturing Technology* 70.9, pp. 1955–1961.

Adra, S. F., A. I. Hamody, I. Griffin, and P. J. Fleming (2005). "A hybrid multi-objective evolutionary algorithm using an inverse neural network for aircraft control system design." In: *Congress on Evolutionary Computation*. IEEE, pp. 1–8.

Affolter, K., T. Hanne, D. Schweizer, and R. Dornberger (2016). "Invasive weed optimization for solving index tracking problems". In: *Soft Computing* 20.9, pp. 3393–3401.

Agyei-Ampomah, S., D. Gounopoulos, and K. Mazouz (2014). "Does gold offer a better protection against losses in sovereign debt bonds than other metals?" In: *Journal of Banking and Finance* 40.1, pp. 507–521.

Al-Anzi, F. and A. Allahverdi (2007). "A self-adaptive differential evolution heuristic for two-stage assembly scheduling problem to minimize maximum lateness with setup times". In: *European Journal of Operational Research* 182.1, pp. 80–94.

— (2013). "An artificial immune system heuristic for two-stage multi-machine assembly scheduling problem to minimize total completion time". In: *Journal of Manufacturing Systems* 32.4, pp. 825–830.

Al Kattan, I. and R. Maragoud (2008). "Performance analysis of flowshop scheduling using genetic algorithm enhanced with simulation". In: *International Journal of Industrial Engineering: Theory, Applications and Practice* 15.1, pp. 62–72.

Al-Salem, A. (2004). "A heuristic to minimize makespan in proportional parallel flow shops". In: *International Journal of Computing & Information Sciences* 2.2, pp. 98–107.

Alexander, C. and A. Dimitriu (2004). "Equity indexing: Optimize your passive investments". In: *Quantitative Finance* 4.3, pp. 30–33.

Allaoui, H., S. Lamouri, and M. Lebbar (2006). "A robustness framework for a stochastic hybrid flow shop to minimize the makespan". In: *International Conference on Service Systems and Service Management*. Vol. 2. IEEE, pp. 1097–1102.

Alshraideh, H. and H. Abu Qdais (2016). "Stochastic modeling and optimization of medical waste collection in Northern Jordan". In: *Journal of Material Cycles and Waste Management*, pp. 1–11.

Amihud, Y. (1996). "Unexpected inflation and stock returns revisited–evidence from Israel". In: *Journal of Money, Credit and Banking* 28.1, pp. 22–33.

Andriosopoulos, K., M. Doumpos, N. C. Papapostolou, and P. K. Pouliasis (2013). "Portfolio optimization and index tracking for the shipping stock and freight markets using evolutionary algorithms". In: *Transportation Research Part E: Logistics and Transportation Review* 52, pp. 16–34.

Antonakakis, N. and R. Kizys (2015). *Dynamic spillovers between commodity and currency markets*.

Armañanzas, R. and J. A. Lozano (2005). "A multiobjective approach to the portfolio optimization problem". In: *IEEE Congress on Evolutionary Computation*. Vol. 2. IEEE Press, pp. 1388–1395.

Asta, S. and E. Ozcan (2014). "An apprenticeship learning hyper-heuristic for vehicle routing in HyFlex". In: *2014 IEEE Symposium on Evolving and Autonomous Learning Systems (EALS)*, pp. 65–72.

Auger, A. and N. Hansen (2005). "Performance evaluation of an advanced local search evolutionary algorithm". In: *2005 IEEE congress on evolutionary computation*. Vol. 2. IEEE, pp. 1777–1784.

Ayodele, A. A. and K. A. Charles (2015). "Portfolio selection problem using generalized differential evolution 3". In: *Applied Mathematical Sciences* 9.42, pp. 2069–2082.

Babaei, S., M. M. Sepehri, and E. Babaei (2015). "Multi-objective portfolio optimization considering the dependence structure of asset returns". In: *European Journal of Operational Research* 244.2, pp. 525–539.

Back, B., T. Laitinen, and K. Sere (1996). "Neural networks and genetic algorithms for bankruptcy predictions". In: *Expert Systems with Applications* 11.4, pp. 407–413.

Bae, S.-T., H.S. Hwang, G.-S. Cho, and M.-J. Goan (2007). "Integrated GA-VRP solver for multi-depot system". In: *Computers & Industrial Engineering* 53, pp. 233–240.

Baker, K. R. and D. Altheimer (2012). "Heuristic solution methods for the stochastic flow shop problem". In: *European Journal of Operational Research* 216.1, pp. 172–177.

Baker, K. R. and D. Trietsch (2011). "Three heuristic procedures for the stochastic, two-machine flow shop problem". In: *Journal of Scheduling* 14.5, pp. 445–454.

Baker, Kenneth R (1974). *Introduction to sequencing and scheduling*. John Wiley & Sons.

Bansal, Y., S. Kumar, and P. Verma (2014). "Commodity futures in portfolio diversification: Impact on investor's utility". In: *Global Business & Management Research* 6.2, pp. 112–121.

Banu, P. K. Nizar and S. Andrews (2015). "Gene clustering using metaheuristic optimization algorithms". In: *International Journal of Applied Metaheuristic Computing* 6.4, pp. 14–38.

Baptista, S., R. C. Oliveira, and E. Zúquete (2002). "A period vehicle routing case study". In: *European Journal of Operational Research* 139, pp. 220–229.

Barreto, S., C. Ferreira, J. Paixao, and B. Sousa (2007). "Using clustering analysis in a capacitated location-routing problem". In: *Eur. J. Oper. Res.* 179.3, pp. 968–977.

Bartz-Beielstein, T., K. E. Parsopoulos, and M. N. Vrahatis (2004). "Design and analysis of optimization algorithms using computational statistics". In: *Applied Numerical Analysis & Computational Mathematics* 1.2, pp. 413–433.

Bastian, C. and A. H. G. R. Kan (1992). "The stochastic vehicle routing problem revisited". In: *European Journal of Operational Research* 56, pp. 407–412.

Battiti, R. and M. Brunato (2005). *Reactive search: machine learning for memory-based heuristics*. Tech. rep. Teofilo F. Gonzalez (Ed.), Approximation Algorithms and Metaheuristics, Taylor & Francis Books (CRC Press.

— (2010). "Reactive search optimization: learning while optimizing". In: *Handbook of Metaheuristics*. Springer, pp. 543–571.

Battiti, R. and G. Tecchiolli (1994). "The reactive tabu search". In: *ORSA journal on computing* 6.2, pp. 126–140.

Bautista, J., E. Fernández, and J. Pereira (2008). "Solving an urban waste collection problem using ants heuristics". In: *Computers & Operations Research* 35, pp. 3020–3033.

Beasley, J. E. (2013). "Portfolio optimisation: Models and solution approaches". In: *Tutorials in operations research* 10, pp. 201–221.

Beasley, J. E., N. Meade, and T.-J. Chang (2003). "An evolutionary heuristic for the index tracking problem". In: *European Journal of Operational Research* 148.3, pp. 621–643.

Bektaş, T. and G. Laporte (2011). "The pollution-routing problem". In: *Transportation Research Part B: Methodological* 45.8, pp. 1232–1250.

Beliën, J., L. De Boeck, and J. Van Ackere (2014). "Municipal solid waste collection and management problems: a literature review". In: *Transportation Science* 48, pp. 78–102.

Belousova, J. and G. Dorfleitner (2012). "On the diversification benefits of commodities from the perspective of euro investors". In: *Journal of Banking and Finance* 36.9, pp. 2455–2472.

Beltrami, E. J. and L. D. Bodin (1974). "Networks and vehicle routing for municipal waste collection". In: *Networks* 4, pp. 65–94.

Benjamin, A. M. and J. E. Beasley (2010). "Metaheuristics for the waste collection vehicle routing problem with time windows, driver rest period and multiple disposal facilities". In: *Computers & Operations Research* Vol. 37, pp. 2270–2280.

Berberoğlu, A. and A. Uyar (2010). "A hyper-heuristic approach for the unit commitment problem". In: *European Conference on the Applications of Evolutionary Computation*. Springer, pp. 121–130.

Berry, K. (2009). "Distributed and Grid Computing via the browser". In:

Bertsimas, D. J. (1988). "Probabilistic combinatorial optimization problems". PhD thesis. Operations Research Center, Massachusetts Institute of Technology.

Bianchi, L., M. Birattari, M. Chiarandini, M. Manfrin, M. Mastrolilli, L. Paquete, O. Rossi-Doria, and T. Schiavinotto (2006). "Hybrid metaheuristics for the vehicle routing problem with stochastic demands". In: *Journal of Mathematical Modelling and Algorithms* 5, pp. 91–110.

Bianchi, L., D. Marco, L. M. Gambardella, and W. J. Gutjahr (2009). "A survey on metaheuristics for stochastic combinatorial optimization". In: *Natural Computing* 8, pp. 239–287.

Bienstock, D. (1996). "Computational study of a family of mixed-integer quadratic programming problems". In: *Mathematical programming* 74.2, pp. 121–140.

Birattari, M. and J. Kacprzyk (2009). *Tuning metaheuristics: a machine learning perspective*. Vol. 197. Springer.

Birattari, M., Z. Yuan, P. Balaprakash, and T. Stützle (2010). "F-Race and iterated F-Race: an overview". In: *Experimental methods for the analysis of optimization algorithms*. Springer, pp. 311–336.

Borshchev, A. and A. Filippov (2004). "From system dynamics and discrete event to practical agent based modeling: reasons, techniques, tools". In: *Proceedings of the 22nd international conference of the system dynamics society*. Vol. 22.

Boussaïd, I., J. Lepagnot, and P. Siarry (2013). "A survey on optimization metaheuristics". In: *Information Sciences* 237, pp. 82–117.

Branch and Cut. http://www.coin-or.org/SYMPHONY/branchandcut/VRP/data/. [Online; accessed March 2016].

Brasileiro, R. C., V. L. F. Souza, B. J. T. Fernandes, and A. L. I. Oliveira (2013). "Automatic method for stock trading combining technical analysis and the artificial bee colony algorithm". In: *2013 IEEE Congress on Evolutionary Computation*, pp. 1810–1817.

Brooks, C. and M. Prokopczuk (2013). "The dynamics of commodity prices". In: *Quantitative Finance* 13.4, pp. 527–542.

Brownlee, A. E. I., O. Regnier-Coudert, J. A. W. McCall, and S. Massie (2010). "Using a Markov network as a surrogate fitness function in a genetic algorithm". In: *IEEE Congress on Evolutionary Computation*. IEEE, pp. 1–8.

Buhrkal, K., A. Larsen, and S. Ropke (2012). "The waste collection vehicle routing problem with time windows in a city logistics context". In: *Procedia - Social and Behavioral Sciences* 39, pp. 241–254.

Burke, E. K., M. Hyde, G. Kendall, G. Ochoa, E. Özcan, and J. R. Woodward (2010). "A classification of hyper-heuristic approaches". In: *Handbook of metaheuristics*, pp. 449–468.

Burke, E. K., M. Gendreau, M. Hyde, G. Kendall, G. Ochoa, E. Özcan, and R. Qu (2013). "Hyper-heuristics: a survey of the state of the art". In: *Eur. J. Oper. Res.* 64.12, pp. 1695–1724.

Cabrera, G., A. A. Juan, D. Lázaro, J. M. Marquès, and I. Proskurnia (2014). "A simulation-optimization approach to deploy Internet services in large-scale systems with user-provided resources". In: *Simulation* 90.6, pp. 644–659.

Caceres, J., P. Arias, D. Guimarans, D. Riera, and A. A. Juan (2014). "Rich vehicle routing problem: a survey". In: *ACM Computing Surveys* 47.2. ISSN: 0360-0300.

Cadenas, J. M., M. C. Garrido, and E. Muñoz (2009). "Using machine learning in a cooperative hybrid parallel strategy of metaheuristics". In: *Inform. Sciences* 179.19, pp. 3255–3267.

Calvet, L. (2015). "Neural networks for routing problems: review and challenges". Green COOP-CYTED Workshop. Madrid, Spain.

Calvet, L. and A. A. Juan (2015). "Educational Data Mining and e-Learning Analytics: an overview of goals, quantitative methods, and time-line evolution". In: *Int. J. of Educational Technology in Higher Education* 12.3.

Calvet, L., A. Ferrer, A. A. Juan, and I. Gomes (2015a). "Market segmentation issues in the multi-depot vehicle routing problem". In: *EURO 2015*. Glasgow, UK.

Calvet, L., A. A. Juan, and N. Schefers (2015b). "Solving the multi-depot vehicle routing problem considering uncertainty and risk factors". In: *Proceedings of the ICRA6 / Risk 2015 Int. Conference*. Barcelona, Spain, pp. 187–194.

Calvet, L., A. A. Juan, and C. Serrat (2015c). "Técnicas estadísticas aplicadas a la calibración de parámetros de metaheurísticas". In: *Proceedings of the X Congreso Español de Metaheurísticas, Algoritmos Evolutivos y Bioinspirados*. Mérida, Spain, pp. 409–416.

Calvet, L., A. Pages, O. Travesset, and A. A. Juan (2016a). "A simheuristic for the heterogeneous site-dependent asymmetric VRP with stochastic demands". In: *Springer Lecture Notes in Computer Science / LNAI*. Vol. 9868, pp. 408–417.

Calvet, L., A. A. Juan, C. Serrat, and J. Ries (2016b). "A statistical learning based approach for parameter fine-tuning of metaheuristics". In: *Statistics and Operations Research Transactions* 40.1, pp. 201–224.

Calvet, L., V. Fernandez-Viagas, J. Framinan, and A. A. Juan (2016c). "Combining simulation with metaheuristics in distributed scheduling problems with stochastic processing times". In: *Proceedings of the 2016 Winter Simulation Conference*. Washington D. C., USA, pp. 2347–2357.

Calvet, L., A. Ferrer, I. Gomes, A. A. Juan, and D. Masip (2016d). "Combining statistical learning with metaheuristics for the multi-depot vehicle routing problem with market segmentation". In: *Computers and Industrial Engineering* 94, pp. 93–104.

Calvet, L., M. Mateo, A. A. Juan, and C. Laroque (2016e). "Optimizing starting times in parallel multiple production lines with stochastic processing times and a shared deadline". In: *Proceedings of the 15th International Conference on Project Management and Scheduling*. Valencia, Spain.

Calvet, L., J. Doering, R. Kizys, A. A. Juan, and A. Fito (2016f). "The stochastic portfolio optimization problem: a formulation and a hybrid methodology". In: *OR58 Annual Conference*. Portsmouth, UK, pp. 127–128.

Calvet, L., J. De Armas, D. Masip, and A. A. Juan (2017). "Learnheuristics: Hybridizing metaheuristics with machine learning for optimization problems with solution-dependent inputs". In: *Open Mathematics*.

Calvet, L., D. Wang, and A. A. Juan (submitted[a]). "A simheuristic algorithm for the stochastic multi-depot vehicle routing problem". In: *International Transactions in Operational Research*.

Calvet, L., R. Kizys, A. A. Juan, and J. Doering (submitted[b]). "A VNS-based simheuristic methodology for the stochastic portfolio optimization problem". In: *Journal of the Operational Research Society*.

Calvet, L., M. Lopeman, J. De Armas, G. Franco, and A. A. Juan (submitted[c]). "Statistical and machine learning approaches for the minimization of trigger errors in earthquake catastrophe bonds". In: *Statistics and Operations Research Transactions*.

Campbell, H. G., R. A. Dudek, and M. L. Smith (1970). "A heuristic algorithm for the n job, m machine sequencing problem". In: *Management science* 16.10, B630–B637.

Canakgoz, N. A. and J. E. Beasley (2009). "Mixed-integer programming approaches for index tracking and enhanced indexation". In: *European Journal of Operational Research* 196.1, pp. 384–399.

Cao, D. and M. Chen (2003). "Parallel flowshop scheduling using Tabu search". In: *International Journal of Production Research* 41.13, pp. 3059–3073.

Carazo, A. F., T. Gómez, J. Molina, A. G. Hernández-Díaz, F. M. Guerrero, and R. Caballero (2010). "Solving a comprehensive model for multiobjective project portfolio selection". In: *Computers & operations research* 37.4, pp. 630–639.

Carvalho, A. R., F. M. Ramos, and A. A. Chaves (2011). "Metaheuristics for the feedforward artificial neural network architecture optimization problem". In: *Neural Computing and Applications* 20.8, pp. 1273–1284.

Caserta, M. and E. Quiñonez Rico (2009). "A cross entropy-Lagrangean hybrid algorithm for the multi-item capacitated lot-sizing problem with setup times". In: *Computers & OR* 36.2, pp. 530–548.

Ceberio, J., E. Irurozki, A. Mendiburu, and J. A. Lozano (2012). "A review on estimation of distribution algorithms in permutation-based combinatorial optimization problems". In: *Pattern Recogn.* 1.1, pp. 103–117.

Cesarone, F., A. Scozzari, and F. Tardella (2013). "A new method for mean-variance portfolio optimization with cardinality constraints". In: *Annals of Operations Research* 205.1, pp. 213–234.

Chan, F. T.S., S.H. Chung, and P.L.Y. Chan (2005). "An adaptive genetic algorithm with dominated genes for distributed scheduling problems". In: *Expert Systems with Applications* 29.2, pp. 364–371.

Chan, Y., W.B. Carter, and M.D. Burnes (2001). "A hybrid algorithm for multi-depot vehicle routing problem". In: *Computers & Operations Research* 28.8, pp. 803–826.

Chang, T.-J., N. Meade, J. E. Beasley, and Y. M. Sharaiha (2000). "Heuristics for cardinality constrained portfolio optimisation". In: *Computers & Operations Research* 27.13, pp. 1271–1302.

ChangYoon, L. and K. Way (2001). "Reliability optimization design using a hybridized genetic algorithm with a neural-network technique". In: *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences* 84.2, pp. 627–637.

Chao, I., B. Golden, and E. Wasil (1993). "A new heuristic for the multi-depot vehicle routing problem that improves upon best known solutions". In: *American Journal of Mathematical and Management Sciences* 13.3-4, pp. 371–406.

Chao, I. M. and T. S. Liou (2005). "A new tabu search heuristic for the site-dependent vehicle routing problem". In: *The Next Wave in Computing, Optimization, and Decision Technologies* 29, pp. 107–119.

Chapelle, O., B. Schölkopf, and A. Zien (2006). *Semi-supervised learning. Adaptive computation and machine learning series*.

Chavarnakul, T. and D. Enke (2009). "A hybrid stock trading system for intelligent technical analysis-based equivolume charting". In: *Neurocomputing* 72.16, pp. 3517–3528.

Chen, L.-W., P. Sharma, and Y.-C. Tseng (2013). "Dynamic traffic control with fairness and throughput optimization using vehicular communications". In: *IEEE Journal on Selected Areas in Communications* 31.9, pp. 504–512.

Chen, M.-Y. (2011). "A hybrid model for business failure prediction-utilization of particle swarm optimization and support vector machines". In: *Neural Network World* 21.2, p. 129.

Chen, P. and X. Xu (2008). "A hybrid algorithm for multi-depot vehicle routing problem". In: *IEEE International Conference on Service Operations and Logistics, and Informatics*, pp. 2031–2034.

Cheung, C. S. and P. Miu (2010). "Diversification benefits of commodity futures". In: *Journal of International Financial Markets, Institutions and Money* 20.5, pp. 451–474.

Chi, B.-W. and C.-C. Hsu (2012). "A hybrid approach to integrate genetic algorithm into dual scoring model in enhancing the performance of credit scoring model". In: *Expert Systems with Applications* 39.3, pp. 2650–2661.

Chiam, S. C., K. C. Tan, and A. Al Mamun (2013). "Dynamic index tracking via multi-objective evolutionary algorithm". In: *Applied Soft Computing* 13.7, pp. 3392–3408.

Chibeles-Martins, N., T. Pinto-Varela, A. P. Barbosa-Póvoa, and A. Q. Novais (2016). "A multi-objective meta-heuristic approach for the design and planning of green supply chains". In: *Expert Systems with Applications* 47, pp. 71–84.

Chica, M., A. A. Juan, O. Cordon, and W. D. Kelton (submitted). "Why Simheuristics?: Benefits, limitations, and best practices when combining metaheuristics with simulation". In: *Simulation Modelling Practice and Theory*, pp. –.

Choi, S. H. and K. Wang (2012). "Flexible flow shop scheduling with stochastic processing times: a decomposition-based approach". In: *Computers & Industrial Engineering* 63.2, pp. 362–373.

Chong, J. and J. Miffre (2010). "Conditional correlation and volatility in commodity futures and traditional asset markets". In: *Journal of Alternative Investments* 12.3, pp. 61–75.

Chorafas, D. N. (2007). "What is meant by risk management?" In: *Risk Management Technology in Financial Services*, pp. 24–42.

Chow, G., E. Jacquier, M. Kritzman, and K. Lowry (1999). "Optimal portfolios in good times and bad". In: *Financial Analysts Journal* 55.3, pp. 65–73.

Christofides, N. and J. E. Beasley (1984). "The period routing problem". In: *Networks* 14, pp. 237–256.

Chua, J. H., G. Sick, and R. S. Woodward (1990). "Diversifying with gold stocks". In: *Financial Analysts Journal* 46.4, pp. 76–79.

Ciner, C., C. Gurdgiev, and B. M. Lucey (2013). "Hedges and safe havens: An examination of stocks, bonds, gold, oil and exchange rates". In: *International Review of Financial Analysis* 29, pp. 202–211.

Clarke, G. and J. Wright (1964). "Scheduling of vehicles from a central depot to a number of delivering points". In: *Operations Research* 12, pp. 568–581.

Cordeau, J. F. and G. Laporte (2001). "A tabu search algorithm for the site dependent vehicle routing problem with time windows". In: *Information Systems and Operational Research* 39.3, pp. 292–298.

Cordeau, J.F. and M. Maischberger (2012). "A parallel iterated tabu search heuristic for vehicle routing problems". In: *Computers & Operations Research* 39.9, pp. 2033–2050.

Cordeau, J.F., M. Gendreau, and G. Laporte (1997). "A tabu search heuristic for periodic and multidepot vehicle routing problems". In: *Networks* 30.2, pp. 105–119.

Corne, D., C. Dhaenens, and L. Jourdan (2012). "Synergies between operations research and data mining: The emerging use of multi-objective approaches". In: *Eur. J. Oper. Res.* 221.3, pp. 469–479.

Cornillier, F., F. Boctor, and J Renaud (2012). "Heuristics for the multi-depot petrol station replenishment problem with time windows". In: *European Journal of Operational Research* 220, pp. 361–369.

Coulouris, G. F., J. Dollimore, and T. Kindberg (2005). *Distributed systems: concepts and design*. Pearson education.

Coy, S. P., B. L. Golden, G. C. Runger, and E. A. Wasil (2001). "Using experimental design to find effective parameter settings for heuristics". In: *Journal of Heuristics* 7.1, pp. 77–97.

Crevier, B., J. F. Cordeau, and G. Laporte (2007). "The multi-depot vehicle routing problem with inter-depot routes". In: *European Journal of Operational Research* 176, pp. 756–773.

Crowston, W. B., F. Glover, and J. D. Trawick (1963). *Probabilistic and parametric learning combinations of local job shop scheduling rules*. Tech. rep. DTIC Document.

Cruz, L., E. Fernandez, C. Gomez, G. Rivera, and F. Perez (2014). "Many-objective portfolio optimization of interdependent projects with 'a priori'incorporation of decision-maker preferences". In: *Appl. Math* 8.4, pp. 1517–1531.

Cura, T. (2009). "Particle swarm optimization approach to portfolio optimization". In: *Nonlinear Analysis: Real World Applications* 10.4, pp. 2396–2406.

Czarn, A., C. MacNish, K. Vijayan, B. Turlach, and R. Gupta (2004). "Statistical exploratory analysis of genetic algorithms". In: *Evolutionary Computation, IEEE Transactions on* 8.4, pp. 405–421.

Dalboni, F. L., L. S. Ochi, and L.M.A. Drummond (2003). "On improving evolutionary algorithms by using data mining for the oil collector vehicle routing problem". In: *International Network Optimization Conference*, pp. 182–188.

Danenas, P. and G. Garsva (2015). "Selection of support vector machines based classifiers for credit risk domain". In: *Expert Systems with Applications* 42.6, pp. 3194–3204.

Daskalaki, C. and G. Skiadopoulos (2011). "Should investors include commodities in their portfolios after all? New evidence". In: *Journal of Banking and Finance* 35.10, pp. 2606–2626.

Davalos, S., F. Leng, E. H. Feroz, and Z. Cao (2014). "Designing an if–then rules-based ensemble of heterogeneous bankruptcy classifiers: A genetic algorithm approach". In: *Intelligent Systems in Accounting, Finance and Management* 21.3, pp. 129–153.

David, W.H.E., A. Kusiak, and A. Artiba (1996). "A scheduling problem in glass manufacturing". In: *IIE Transactions (Institute of Industrial Engineers)* 28.2, pp. 129–139.

De Armas, J., L. Calvet, G. Franco, M. Lopeman, and A. A. Juan (2016a). "Minimizing trigger error in parametric earthquake catastrophe bonds via statistical approaches". In: *Springer Lecture Notes in Business Information Processing* 254, pp. 167–175.

De Castro, L. N. and F. J. Von Zuben (2002). "Learning and optimization using the clonal selection principle". In: *IEEE transactions on evolutionary computation* 6.3, pp. 239–251.

De Jong, K. (2007). "Parameter setting in EAs: a 30 year perspective". In: *Parameter setting in evolutionary algorithms*. Springer, pp. 1–18.

De Lima, Francisco Chagas, Jorge Dantas De Melo, and Adrião Duarte Doria Neto (2008). "Using the Q-learning algorithm in the constructive phase of the GRASP and reactive GRASP metaheuristics". In: pp. 4169–4176. ISBN: 9781424418213.

Deb, K., S. Bandaru, D. Greiner, A. Gaspar-Cunha, and C. C. Tutum (2014). "An integrated approach to automated innovization for discovering useful design principles: Case studies from engineering". In: *Applied Soft Computing* 15, pp. 42–56.

Delucchi, M. A. and D. R. McCubbin (2010). *External costs of transport in the US*. Tech. rep.

Demir, E., T. Van Woensel, and T. de Kok (2014). "Multidepot distribution planning at logistics service provider Nabuurs BV". In: *Interfaces* 44.6, pp. 591–604.

Deng, G.-F., W.-T. Lin, and C.-C. Lo (2012). "Markowitz-based portfolio selection with cardinality constraints using improved particle swarm optimization". In: *Expert Systems with Applications* 39.4, pp. 4558–4566.

Derigs, U. and N. Nickel (2003). "Meta-heuristic based decision support for portfolio optimization with a case study on tracking error minimization in passive portfolio management". In: *OR Spectrum* 25.3, pp. 345–378.

Derigs, U. and N. H. Nickel (2004). "On a local-search heuristic for a class of tracking error minimization problems in portfolio management". In: *Annals of Operations Research* 131.1-4, pp. 45–77.

Dhaenens, C. and L. Jourdan (2016). *Metaheuristics for Big Data*. Wiley. ISBN: 9781119347606.

Di Gaspero, L., G. Di Tollo, A. Roli, and A. Schaerf (2011). "Hybrid metaheuristics for constrained portfolio selection problems". In: *Quantitative Finance* 11.10, pp. 1473–1487.

Di Tollo, G. and A. Roli (2008). "Metaheuristics for the portfolio selection problem". In: *International Journal of Operations Research* 5.1, pp. 13–35.

Díaz-Manríquez, A., G. Toscano-Pulido, and W. Gómez-Flores (2011). "On the selection of surrogate models in evolutionary optimization algorithms". In: *2011 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, pp. 2155–2162.

Dobslaw, F. (2010). "A parameter tuning framework for metaheuristics based on design of experiments and artificial neural networks". In: *World Academy of Science, Engineering and Technology* 64, pp. 213–216.

Doering, J., L. Calvet, R. Kizys, A. Fito, and A. A. Juan (2016a). "Rich portfolio optimization with stocks and individual commodity futures Contracts". In: *Portsmouth-Fordham Conference on Banking & Finance. Portsmouth, UK*.

Doering, J., A. A. Juan, R. Kizys, A. Fito, and L. Calvet (2016b). "Solving realistic portfolio optimization problems via metaheuristics: a survey and an example". In: *Springer Lecture Notes in Business Information Processing* 254, pp. 22–30.

Doering, J., L. Calvet, A. Fito, R. Kizys, and A. A. Juan (submitted[a]). "Metaheuristics for realistic portfolio optimization and risk management: current state and future trends". In: *Annals of Operations Research*.

Doering, J., L. Calvet, R. Kizys, A. Fito, and A. A. Juan (submitted[b]). "Rich portfolio optimization with stocks and individual commodity futures contracts". In: *OR Spectrum*.

Doerner, K., W. J. Gutjahr, R. F. Hartl, C. Strauss, and C. Stummer (2004). "Pareto ant colony optimization: A metaheuristic approach to multiobjective portfolio selection". In: *Annals of operations research* 131.1-4, pp. 79–99.

Doerner, K. F., W. J. Gutjahr, R. F. Hartl, C. Strauss, and C. Stummer (2006). "Pareto ant colony optimization with ILP preprocessing in multiobjective project portfolio selection". In: *European Journal of Operational Research* 171.3, pp. 830–841.

Dominguez, O., D. Guimarans, A. A. Juan, and I. de la Nuez (2016a). "A Biased-Randomised Large Neighbourhood Search for the two-dimensional Vehicle Routing Problem with Backhauls". In: *European Journal of Operational Research*.

Dominguez, O., A. A. Juan, J. Barrios B.and Faulin, and A. Agustin (2016b). "Using biased randomization for solving the two-dimensional loading vehicle routing problem with heterogeneous fleet". In: *Annals of Operations Research* 236.2, pp. 383–404.

Dondo, R. and J. Cerdá (2009). "A hybrid local improvement algorithm for large-scale multi-depot vehicle routing problems with time windows". In: *Computers & Chemical Engineering* 33, pp. 513–530.

Dorigo, M. (1992). "Optimization, learning and natural algorithms". PhD thesis. Politecnico di Milano, Italy.

Dror, M. and P. Trudeau (1986). "Stochastic vehicle routing with modified savings algorithms". In: *European Journal of Operational Research* 23, pp. 228–235.

Eiben, A. E., R. Hinterding, and Z. Michalewicz (1999). "Parameter control in evolutionary algorithms". In: *Evolutionary Computation, IEEE Transactions on* 3.2, pp. 124–141.

Escalante, H. J., V. Ponce-López, S. Escalera, X. Baró, A. Morales-Reyes, and J. Martínez-Carranza (2016). "Evolving weighting schemes for the bag of visual words". In: *Neural Comput. Appl.* 0, pp. 1–15.

Escobar, J. W., R. Linfati, P. Toth, and M. G. Baldoquin (2014). "A hybrid granular tabu search algorithm for the multi-depot vehicle routing problem". In: *Journal of Heuristics* 20, pp. 483–509.

Euchi, J. (2014). "Hybrid estimation of distribution algorithm for a multiple trips fixed fleet vehicle routing problems with time windows". In: *International Journal of Operational Research* 21.4, p. 433. ISSN: 1745-7645.

European Commission (2015). *EU Transport in figures*. Statistical pocketbook.

European Union (2011a). *Eurostat regional yearbook 2014. Statistical books*. Tech. rep. Publications Office of the European Union.

— (2011b). *Roadmap to a single European Transport Area – towards a competitive and resource-efficient transport system*. Tech. rep. Publications Office of the European Union.

Eurostat (2015). *Sustainable development in the European Union: 2015 monitoring report of the EU sustainable development strategy*. Publications office of the European Union.

Evans, C., K. Pappas, and F. Xhafa (2013). "Utilizing artificial neural networks and genetic algorithms to build an algo-trading model for intra-day foreign exchange speculation". In: *Mathematical and Computer Modelling* 58.5, pp. 1249–1266.

Fama, E. F. (1981). "Stock returns, real activity, inflation, and money". In: *The American Economic Review* 71.4, pp. 545–565.

Farmer, J. D., N. H. Packard, and A. S. Perelson (1986). "The immune system, adaptation, and machine learning". In: *Phys. D* 2.1-3, pp. 187–204.

Faulin, J., A. Juan, F. Lera, and S. Grasman (2011). "Solving the capacitated vehicle routing problem with environmental criteria based on real estimations in road transportation: a case study". In: *Procedia-Social and Behavioral Sciences* 20, pp. 323–334.

Faulin, J., A. A. Juan, S. E. Grasman, and M. J. Fry (2012). *Decision making in service industries: A practical approach*. CRC Press.

Feo, T. A. and M. G. C. Resende (1989). "A probabilistic heuristic for a computationally difficult set covering problem". In: *Oper. Res. Lett.* 8.2, pp. 67–71.

Feo, T. A and M. G. C. Resende (1995). "Greedy randomized adaptive search procedures". In: *Journal of global optimization* 6.2, pp. 109–133.

Fernández, A. and S. Gómez (2007). "Portfolio selection using neural networks". In: *Computers and Operations Research* 34.4, pp. 1177–1191.

Fernandez, E., C. Gomez, G. Rivera, and L. Cruz-Reyes (2015). "Hybrid metaheuristic approach for handling many objectives and decisions on partial support in project portfolio optimisation". In: *Information Sciences* 315, pp. 102–122.

Fernández-Caballero, J. C., F. J. Martinez, C. Hervas, and P. A. Gutierrez (2010). "Sensitivity versus accuracy in multiclass problems using memetic pareto evolutionary neural networks". In: *IEEE T. Neural Network.* 21.5, pp. 750–770.

Fernandez-Viagas, V. and J. M. Framinan (2015c). "NEH-based heuristics for the permutation flowshop scheduling problem to minimise total tardiness". In: *Computers & Operations Research* 60, 27—36.

Ferone, D., A. Facchiano, A. Marabotti, and P. Festa (2016). "A new GRASP metaheuristic for biclustering of gene expression data". In: *PeerJ PrePrints*.

Figueira, G., M. Furlan, and B. Almada-Lobo (2013). "Predictive production planning in an integrated pulp and paper mill". In: *Manufacturing Modelling, Management, and Control*. Vol. 7. 1, pp. 371–376.

Fikar, C., A. A. Juan, E. Martinez, and P. Hirsch (2016). "A discrete-event driven metaheuristic for dynamic home service routing with synchronised trip sharing". In: *European Journal of Industrial Engineering* 10, pp. 323–340.

Framinan, J. M. and P. Perez-Gonzalez (2015). "On heuristic solutions for the stochastic flowshop scheduling problem". In: *Journal of Operational Research* 246.2, 413—420.

Freitas, A. (2002). "Data mining and knowledge discovery with evolutionary algorithms". In: *Advances in Evolutionary Computation* 105, pp. 819–845.

— (2008). "A review of evolutionary algorithms for data mining". In: *Soft Computing for Knowledge Discovery and Data Mining*. Springer, pp. 79–111.

Gao, J. and R. Chen (2011). "A hybrid genetic algorithm for the distributed permutation flowshop scheduling problem". In: *International Journal of Computational Intelligence Systems* 4.4, pp. 497–508.

Gao, J., R. Chen, and W. Deng (2013). "An efficient tabu search algorithm for the distributed permutation flowshop scheduling problem". In: *International Journal of Production Research* 51.3, pp. 641–651.

García, V., A. I. Marqués, and J. S. Sánchez (2015). "An insight into the experimental design for credit risk and corporate bankruptcy prediction systems". In: *Journal of Intelligent Information Systems* 44.1, pp. 159–189.

Gardi, F., T. Benoist, J. Darlay, B. Estellon, and R. Megel (2014). *Mathematical programming solver based on local search*. John Wiley & Sons.

Garey, M. R., D. S. Johnson, and R. Sethi (1976). "The complexity of flowshop and jobshop scheduling". In: *Mathematics of Operations Research* 1.2, pp. 117–129.

Garrett, I. and N. Taylor (2001). "Portfolio diversification and excess comovement in commodity prices". In: *The Manchester School* 69.4, pp. 351–368.

Gaspar-Cunha, A. and Armando S. Vieira (2004). "A hybrid multi-objective evolutionary algorithm using an inverse neural network". In: pp. 25–30.

Gaspar-Cunha, A., G. Recio, L. Costa, and C. Estébanez (2014). "Self-adaptive MOEA feature selection for classification of bankruptcy prediction data". In: *The Scientific World Journal* 2014.

Gass, S. I. and A. A. Assad (2005). "Model world: tales from the time line—the definition of OR and the origins of Monte Carlo simulation". In: *Interfaces* 35.5, pp. 429–435.

Geman, H. and C. Kharoubi (2008). "WTI crude oil futures in portfolio diversification: the time-to-maturity effect". In: *Journal of Banking and Finance* 32.12, pp. 2553–2559.

Gendreau, M., G. Laporte, and R. Séguin (1995). "An exact algorithm for the vehicle routing problem with stochastic demands and customers". In: *Transportation Science* 29, pp. 143–155.

— (1996). "Stochastic vehicle routing with modified savings algorithms". In: *European Journal of Operational Research* 88, pp. 3–12.

Ghasemzadeh, F. and N. P. Archer (2000). "Project portfolio selection through decision support". In: *Decision support systems* 29.1, pp. 73–88.

Ghezail, F., H. Pierreval, and S. Hajri-Gabouj (2010). "Analysis of robustness in proactive scheduling: a graphical approach". In: *Computers & Industrial Engineering* 58.2, pp. 193–198.

Ghiani, G., F. Guerriero, G. Improta, and R. Musmanno (2005). "Waste collection in Southern Italy: solution of a real-life arc routing problem". In: *International Transactions in Operational Research* 12, pp. 135–144.

Ghiani, G., C. Mourão, L. Pinto, and D. Vigo (2014b). "Routing in waste collection applications". In: *Arc routing: problems, methods, and applications*. Ed. by A. Corberán and G. Laporte. Philadelphia: SIAM Monographs on Discrete Mathematics and Applications, pp. 351–370.

Gillett, B.E. and J.G. Johnson (1976). "Multi-terminal vehicle-dispatch algorithm". In: *Omega* 4.6, pp. 711–718.

Gilli, M., D. Maringer, and E. Schumann (2011). *Numerical methods and optimization in finance*. Academic Press.

Glover, F. (1977). "Heuristics for integer programming using surrogate constraints". In: *Decision Science* 8.1, pp. 156–166.

— (1986). "Future paths for integer programming and links to artificial intelligence". In: *Computers & Operations Research* 13.5, pp. 533–549.

— (1989). "Tabu search—part I". In: *ORSA Journal on computing* 1.3, pp. 190–206.

— (2000). "Multi-start and strategic oscillation methods—principles to exploit adaptive memory". In: *Computing tools for modeling, optimization and simulation*, pp. 1–23.

Göçken, M., M. Özçalıcı, A. Boru, and A. T. Dosdoğru (2016). "Integrating metaheuristics and artificial neural networks for improved stock price prediction". In: *Expert Systems with Applications* 44, pp. 320–331.

Golden, B. L., S. Raghavan, and E. A. Wasil (2008). *The vehicle routing problem: latest advances and new challenges*. Vol. 43. Springer Science & Business Media.

Golden, B.L., T.L. Magnanti, and H.Q. Nguyen (1977). "Implementing vehicle routing algorithms". In: *Networks* 7.2, pp. 113–148.

Golmakani, H. R. and M. Fazel (2011). "Constrained portfolio selection using particle swarm optimization". In: *Expert Systems with Applications* 38.7, pp. 8327–8335.

Gonzalez, S., D. Riera, A. A. Juan, M. G. Elizondo, and P. Fonseca (2012). "Proceedings of the 2012 Winter Simulation Conference". In: *SIM-RANDSHARP: a hybrid algorithm for solving the arc routing problem with stochastic demands*. Ed. by C. Laroque, J. Himmelspach, R. Pasupathy, O. Rose, and A. M. Uhrmacher, pp. 1–12.

Gonzalez, S., A. A. Juan, D. Riera, M. Elizondo, and J. Ramos (2016). "A simheuristic algorithm for solving the arc routing problem with stochastic demands". In: *Journal of Simulation*.

Goodson, J. C. (2015). "A priori policy evaluation and cyclic-order-based simulated annealing for the multi-compartment vehicle routing problem with stochastic demands". In: *European Journal of Operational Research* 241, pp. 361–369.

Gordini, N. (2014). "A genetic algorithm approach for SMEs bankruptcy prediction: Empirical evidence from Italy". In: *Expert Systems with Applications* 41.14, pp. 6433–6445.

Gorgulho, A., R. Neves, and N. Horta (2011). "Applying a GA kernel on optimizing technical analysis rules for stock picking and portfolio composition". In: *Expert systems with Applications* 38.11, pp. 14072–14085.

Grasas, A., A. Juan, J. Faulin, J. De Armas, and H. Ramalhinho (submitted). "Biased Randomization of Heuristics using Skewed Probability Distributions: a survey and some applications". In: *Computers and Industrial Engineering*, pp. –.

Gruler, A., A. A. Juan, and M. Steglich (2015b). "Proceedings of the 2015 Metaheuristics International Conference. Agadir, Morocco". In: *A heuristic approach for smart waste collection management*.

Gruler, A., C. Quintero, L. Calvet, and A. A. Juan (2016). "Waste collection under uncertainty: A simheuristic based on variable neighborhood search". In: *European Journal of Industrial Engineering*.

Guastaroba, G. and M. G. Speranza (2012). "Kernel search: An application to the index tracking problem". In: *European Journal of Operational Research* 217.1, pp. 54–68.

Gulczynski, D., B. L. Golden, and E. Wasil (2011). "The multi-depot split delivery vehicle routing problem: An integer programming-based heuristic, new test problems, and computational results". In: *Computers & Industrial Engineering* 61, pp. 794–804.

Gumustekin, S., T. Senel, and M. A. Cengiz (2014). "A comparative study on Bayesian optimization algorithm for nutrition problem". In: *J. Food Nutr. Res.* 2.12, pp. 952–958.

Gunawan, A., H. C. Lau, and E. Wong (2013). "Real-world parameter tuning using factorial design with parameter decomposition". In: *Advances in Metaheuristics*. Springer, pp. 37–59.

Gutjahr, W. J., S. Katzensteiner, P. Reiter, C. Stummer, and M. Denk (2008). "Competence-driven project portfolio selection, scheduling and staff assignment". In: *Central European Journal of Operations Research* 16.3, pp. 281–306.

Gutjahr, W. J., S. Katzensteiner, C. Reiter P.and Stummer, and M. Denk (2010). "Multi-objective decision analysis for competence-oriented project portfolio selection". In: *European Journal of Operational Research* 205.3, pp. 670–679.

Hahn, G. J. and N. Doganaksoy (2012). *A career in statistics: beyond the numbers*. John Wiley & Sons.

Han, H. and E. Ponce-Cueto (2015). "Waste collection vehicle routing problem: literature review". In: *Traffic & Transportation* 27, pp. 345–358.

Hansen, P., Mladenović N., Brimberg J., and Moreno J. A. (2010a). "Variable neighborhood search". In: *Handbook of metaheuristics*. Ed. by Gendreau M. and Potvin J.-Y. 2nd. Springer, pp. 61–86.

Hansen, P., N. Mladenovic, and J. A. Moreno (2010b). "Variable neighbourhood search: methods and applications". In: *Annals of Operations Research* 175.1, pp. 367–407.

Hardouvelis, G. A. (1987). "Macroeconomic information and stock prices". In: *Journal of Economics and Business* 39.2, pp. 131–140.

Hastie, T., R. Tibshirani, and J. Friedman (2009). *The elements of statistical learning*. 2nd ed. Springer.

Hatami, S., R. Ruiz, and C. Andrés-Romano (2013). "The distributed assembly permutation flowshop scheduling problem". In: *International Journal of Production Research* 51.17, pp. 5292–5308.

— (2015). "Heuristics and metaheuristics for the distributed assembly permutation flowshop scheduling problem with sequence dependent setup times". In: *International Journal of Production Economics* 169, pp. 76–88.

Hatami, S., L. Calvet, V. Fernandez-Viagas, J. Framinan, and A. A. Juan (submitted). "Combining simulation with metaheuristics in distributed scheduling problems with stochastic processing times". In: *International Transactions in Operational Research*, pp. –.

He, G. and N. Huang (2012). "A modified particle swarm optimization algorithm with applications". In: *Applied Mathematics and Computation* 219.3, pp. 1053–1060.

— (2014). "A new particle swarm optimization algorithm with an application". In: *Applied Mathematics and Computation* 232, pp. 521–528.

Hemmelmayr, V., Karl F. Doerner, R. F. Hartl, and S. Rath (2013). "A heuristic solution method for node routing based solid waste collection problems". In: *Journal of Heuristics* 19, pp. 129–156.

Hemmelmayr, V., K. F. Doerner, R. F. Hartl, and D. Vigo (2014). "Models and algorithms for the integrated planning of bin allocation and vehicle routing in solid waste management". In: *Transportation Science* 48, pp. 103–120.

Herrero, R., A. Rodriguez, J. Caceres-Cruz, and A. A. Juan (2014). "Solving vehicle routing problems with asymmetric costs and heterogeneous fleets". In: *International Journal of Advanced Operations Management* 6.1, pp. 58–80.

Hillier, D., P. Draper, and R. Faff (2006). *Do precious metals shine? An investment perspective*.

Ho, W., G.T.S. Ho, P. Ji, and H.C.W. Lau (2008). "A hybrid genetic algorithm for the multi-depot vehicle routing problem". In: *Engineering Applications of Artificial Intelligence* 21.4, pp. 548–557.

Hoff, A., H. Andersson, M. Christiansen, G. Hasle, and A. Løkketangen (2010). "Industrial aspects and literature survey: fleet composition and routing". In: *Computers and Operations Research* 37.9, pp. 2041–2061.

Holland, J. H. (1962). "The compact genetic algorithm". In: *Journal of the ACM* 3.9, pp. 297–314.

Hooker, J. N. (1995). "Testing heuristics: We have it all wrong". In: *Journal of Heuristics* 1.1, pp. 33–42.

Hruschka, E. R., R. J. G. B. Campello, and A. A. Freitas (2009). "A survey of evolutionary algorithms for clustering". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 39.2, pp. 133–155.

Hsu, C.-Y., P.-C. Chang, and M.-H. Chen (2015). "A linkage mining in block-based evolutionary algorithm for permutation flowshop scheduling problem". In: *Computers & Industrial Engineering* 83, pp. 159–171.

Hu, X.-B. and X.-Y. Huang (2004). "Solving TSP with characteristic of clustering by ant colony algorithm". In: *Acta Simulata Systematica Sinica* 12, p. 014.

Hutter, F., H. H. Hoos, K. Leyton-Brown, and T. Stützle (2009). "ParamILS: an automatic algorithm configuration framework". In: *Journal of Artificial Intelligence Research* 36.1, pp. 267–306.

Ismail, Z. and I. Irhamah (2008). "Solving the vehicle routing problem with stochastic demands via hybrid genetic algorithm-tabu search". In: *Journal of Mathematics and Statistics* 4, pp. 161–167.

Ismail, Z. and S.L. Loh (2009). "Ant colony optimization for solving solid waste collection scheduling problems". In: *Journal of Mathematics and Statistics* 5, pp. 199–205.

Jabbarpour, M. R., R. Md. Noor, and R. H. Khokhar (2015). "Green vehicle traffic routing system using ant-based algorithm". In: *Journal of Network and Computer Applications* 58, pp. 294–308.

Jaffe, J. F. (1989). "Gold and gold stocks as investments for institutional portfolios". In: *Financial Analysts Journal* 45.2, pp. 53–59.

Jensen, G. R., R. R. Johnson, and J. M. Mercer (2002). "Tactical asset allocation and commodity futures". In: *The Journal of Portfolio Management* 28.4, pp. 100–111.

Jeong, S.-J., K.-S. Kim, and Y.-H. Lee (2009). "The efficient search method of simulated annealing using fuzzy logic controller". In: *Expert Systems with Applications* 36.3, pp. 7099–7103.

Jin, Y. (2005). "A comprehensive survey of fitness approximation in evolutionary computation". In: *Soft. Comput.* 9.1, pp. 3–12.

Jin, Y. and B. Sendhoff (2004). "Reducing fitness evaluations using clustering techniques and neural network ensembles". In: *Genetic and Evolutionary Computation Conference*. Springer, pp. 688–699.

Johnson, D. S. (2002). "A theoretician's guide to the experimental analysis of algorithms". In: *Data structures, near neighbor searches, and methodology: fifth and sixth DIMACS implementation challenges* 59, pp. 215–250.

Johnson, S.M. (1954). "Optimal two- and three-stage production schedules with setup times included". In: *Naval research logistics quarterly* 1, pp. 61–68.

Jourdan, L., D. Corne, D. Savic, and G. Walters (2005). "Preliminary investigation of the learnable evolution model for faster/better multiobjective water systems design". In: *International Conference on Evolutionary Multi-Criterion Optimization*. Springer, pp. 841–855.

Jourdan, L., C. Dhaenens, and E.-G. Talbi (2006). "Using datamining techniques to help metaheuristics: a short survey". In: *International Workshop on Hybrid Metaheuristics*. Springer Berlin Heidelberg. Gran Canaria, Spain, pp. 57–69.

Juan, A. A., J. Faulin, R. Ruiz, B. Barrios, M. Gilibert, and X. Vilajosana (2009c). "Using oriented random search to provide a set of alternative solutions to the capacitated vehicle routing problem". In: *Operations Research and Cyber-Infrastructure*. Springer, New York, USA, pp. 331–346.

Juan, A. A., J. Faulin, R. Ruiz, B. Barrios, and S. Caballe (2010). "The SR-GCWS hybrid algorithm for solving the capacitated vehicle routing problem". In: *Applied Soft Computing* 10.1, pp. 215–224.

Juan, A. A., J. Faulin, J. Jorba, D. Riera, D. Masip, and B. Barrios (2011a). "On the use of Monte Carlo simulation, cache and splitting techniques to improve the Clarke and Wright savings heuristics". In: *Journal of the Operational Research Society* 62, pp. 1085–1097.

Juan, A. A., J. Faulin, S. Grasman, D. Riera, J. Marull, and C. Mendez (2011b). "Using safety stocks and simulation to solve the vehicle routing problem with stochastic demands". In: *Transportation Research Part C: Emerging Technologies* 19.5, pp. 751–765.

Juan, A. A., J. Faulin, A. Ferrer, H. Lourenço, and B. Barrios (2013a). "MIRHA: multi-start biased randomization of heuristics with adaptive local search for solving non-smooth routing problems". In: *TOP* 21, pp. 109–132.

Juan, A. A., J. Faulin, J. Jorba, J. Caceres, and J. M. Marques (2013b). "Using parallel & distributed computing for real-time solving of vehicle routing problems with stochastic demands". In: *Annals of Operations Research* 207.1, pp. 43–65.

Juan, A. A., B. Barrios, E. Vallada, D. Riera, and J. Jorba (2014a). "A simheuristic algorithm for solving the permutation flow shop problem with stochastic processing times". In: *Simulation Modelling Practice and Theory* 46, pp. 101–117.

Juan, A. A., S. E. Grasman, J. Caceres-Cruz, and T. Bektaş (2014b). "A simheuristic algorithm for the single-period stochastic inventory-routing problem with stock-outs". In: *Simulation Modelling Practice and Theory* 46, pp. 40–52.

Juan, A. A., J. Faulin, J. Caceres, B. Barrios, and E. Martinez (2014c). "A successive approximations method for the heterogeneous vehicle routing problem: Analyzing different fleet configurations". In: *European Journal of Industrial Engineering* 8.6, pp. 762–788.

Juan, A. A., J. Cáceres-Cruz, S. Gonzalez, D. Riera, and B. Barrios (2014d). "Biased randomization of classical heuristics". In: *Encyclopedia of Business Analytics and Optimization, IGI Global* 1, pp. 314–324.

Juan, A. A., J. Faulin, S. Grasman, M. Rabe, and G. Figueira (2015a). "A review of simheuristics: extending metaheuristics to deal with stochastic optimization problems". In: *Operations Research Perspectives* 2, pp. 62–72.

Juan, A. A., J. Faulin, L. Calvet, A. Pages, and C. Quintero (2015b). "Applications of simheuristics in transportation and logistics". In: *EURO 2015*. Glasgow, UK.

Juan, A. A., I. Pascual, D. Guimarans, and B. Barrios (2015c). "Combining biased randomization with iterated local search for solving the multidepot vehicle routing problem". In: *International Transactions in Operational Research* 22.4, pp. 647–667.

Juan, A. A., J. Faulin, and L. Calvet (2015d). "Supporting real-time decision-making in logistics and transportation by combining simulation with heuristics". In: *Proceedings of the 12th Int. Multidisciplinary Modeling & Simulation Conf.* Bergeggi, Italy, pp. 35–38.

Kadziński, M., T. Tervonen, M. K. Tomczyk, and R. Dekker (2017). "Evaluation of multi-objective optimization approaches for solving green supply chain design problems". In: *Omega* 68, pp. 168–184.

Kahn, K. B., S. E. Kay, R. J. Slotegraaf, and S. Uban (2013). *The PDMA handbook of new product development*. Third. John Wiley & Sons, Inc. ISBN: 9780470648209.

Kanda, J. Y., A. C. P. L. F. Carvalho, E. R. Hruschka, and C. Soares (2011). "Using meta-learning to recommend meta-heuristics for the traveling salesman problem". In: *Machine Learning and Applications and Workshops (ICMLA), 2011 10th International Conference on*. Vol. 1. IEEE, pp. 346–351.

Karakatic, S. and V. Podgorelec (2015). "A survey of genetic algorithms for solving multi depot vehicle routing problem". In: *Applied Soft Computing* 27, pp. 519–532.

Katragjini, K., E. Vallada, and R. Ruiz (2013). "Flow shop rescheduling under different types of disruption". In: *International Journal of Production Research* 51.3, pp. 780–797.

Kennedy, J. and R. C. Eberhart (1995). "Particle swarm optimization". In: *Proceedings of the IEEE International Conference on Neural Networks*, pp. 1942–1948.

Kenyon, A. S. and D. P. Morton (2003). "Stochastic vehicle routing with random travel times". In: *Transportation Science* 37, pp. 69–82.

Khabzaoui, M, C Dhaenens, and E.-G. Talbi (2004). "A multicriteria genetic algorithm to analyze DNA microarray data". In: *Cec2004: Proceedings of the 2004 Congress on Evolutionary Computation, Vols 1 and 2*, pp. 1874–1881.

Khabzaoui, M., C. Dhaenens, and E.-G. Talbi (2008). "Combining evolutionary algorithms and exact approaches for multi-objective knowledge discovery". In: *RAIRO Operations Research* 42.1, pp. 69–83.

Kianfar, K., S. M. T. F. Ghomi, and A. O. Jadid (2012). "Study of stochastic sequence-dependent flexible flow shop via developing a dispatching rule and a hybrid GA". In: *Engineering Applications of Artificial Intelligence* 25.3, pp. 494–506.

Kim, B., S. Kim, and S. Sahoo (2006). "Waste collection vehicle routing problem with time windows". In: *Computers & Operations Research* Vol. 33, pp. 3624–3642.

Kim, M.-J. and I. Han (2003). "The discovery of experts' decision rules from qualitative bankruptcy data using genetic algorithms". In: *Expert Systems with Applications* 25.4, pp. 637–646.

Kim, M.-J. and D.-K. Kang (2010). "Ensemble with neural networks for bankruptcy prediction". In: *Expert Systems with Applications* 37.4, pp. 3373–3379.

— (2012). "Classifiers selection in ensembles using genetic algorithms for bankruptcy prediction". In: *Expert Systems with applications* 39.10, pp. 9308–9314.

Kirkos, E. (2015). "Assessing methodologies for intelligent bankruptcy prediction". In: *Artificial Intelligence Review* 43.1, pp. 83–123.

Kirkpatrick, S. (1984). "Optimization by simulated annealing: Quantitative studies". In: *Journal of statistical physics* 34.5-6, pp. 975–986.

Kizys, R., A. A. Juan, B. Sawik, and L. Calvet (submitted). "ARPO: an iterated local search algorithm for portfolio optimization under realistic constraints". In: *Quantitative Finance*.

Koç, Ç., T. Bektaş, O. Jabali, and G. Laporte (2014). "The fleet size and mix pollution-routing problem". In: *Transportation Research Part B: Methodological* 70, pp. 239–254.

Kolm, P. N., R. Tütüncü, and F. J. Fabozzi (2014). "60 years of portfolio optimization: Practical challenges and current trends". In: *European Journal of Operational Research* 234.2, pp. 356–371.

Komaki, G.M., E. Teymourian, V. Kayvanfar, and Z. Booyavi (2017). "Improved discrete cuckoo optimization algorithm for the three-stage assembly flowshop scheduling problem". In: *Computers and Industrial Engineering* 105, pp. 158–173.

Koulamas, C. and G. J. Kyparisis (2001). "The three-stage assembly flowshop scheduling problem". In: *Computers & Operations Research* 28.7, pp. 689–704.

Kozeny, V. (2015). "Genetic algorithms for credit scoring: Alternative fitness function performance comparison". In: *Expert Systems with Applications* 42.6, pp. 2998–3004.

Krink, T. and S. Paterlini (2011). "Multiobjective optimization using differential evolution for real-world portfolio optimization". In: *Computational Management Science* 8.1-2, pp. 157–179.

Krink, T., S. Mittnik, and S. Paterlini (2009). "Differential evolution and combinatorial search for constrained index-tracking". In: *Annals of Operations Research* 172.1, pp. 153–176.

Kuhn, M. (2008). "Building predictive models in R using the caret package". In: *Journal of Statistical Software* 28.5.

Kumar, P. R. and V. Ravi (2007). "Bankruptcy prediction in banks and firms via statistical and intelligent techniques–A review". In: *European journal of operational research* 180.1, pp. 1–28.

Kuo, Y. (2010). "Using simulated annealing to minimize fuel consumption for the time-dependent vehicle routing problem". In: *Computers & Industrial Engineering* 59.1, pp. 157–165.

Kurada, R. R., K. K. Pavan, and A. V. Rao (2013). "A preliminary survey on optimized multiobjective metaheuristic methods for data clustering using evolutionary approaches". In: *International Journal of Computer Science & Information Technology* 5.

Lantz, B. (2013). *Machine learning with R*. Packt Publishing.

Laporte, G., F. Louveaux, and H. Mercure (1992). "The vehicle routing problem with stochastic travel times". In: *Transportation Science* 26, pp. 161–170.

Lessmann, S., M. Caserta, and I. M. Arango (2011). "Tuning metaheuristics: a data mining based approach for particle swarm optimization". In: *Expert Systems with Applications* 38.10, pp. 12826–12838.

Leung, Y.-W. and Y. Wang (2001). "An orthogonal genetic algorithm with quantization for global numerical optimization". In: *Trans. Evol. Comp* 5.1, pp. 41–53.

Li, J., E. K. Burke, and R. Qu (2011a). "Integrating neural networks and logistic regression to underpin hyper-heuristic search". In: *Knowl.-based Syst.* 24.2, pp. 322–330.

Li, J., P.M. Pardalos, H. Sun, J. Pei, and Y. Zhang (2015). "Iterated local search embedded adaptive neighborhood selection approach for the multi-depot vehicle routing problem with simultaneous deliveries and pickups". In: *Expert Systems with Applications* 42.7, pp. 3551–3561.

Li, Q. and L. Bao (2014). "Enhanced index tracking with multiple time-scale analysis". In: *Economic Modelling* 39, pp. 282–292.

Liao, C.-J., C.-H. Lee, and H.-C. Lee (2015). "An efficient heuristic for a two-stage assembly scheduling problem with batch setup times to minimize makespan". In: *Computers & Industrial Engineering* 88, pp. 317–325.

Lim, D., Y. Jin, Y.-S. Ong, and B. Sendhoff (2010). "Generalizing Surrogate-assisted Evolutionary Computation". In: *Trans. Evol. Comp* 14.3, pp. 329–355.

Lin, Q., L. Gao, X. Li, and C. Zhang (2015). "A hybrid backtracking search algorithm for permutation flow-shop scheduling problem". In: *Computers & Industrial Engineering* 85, pp. 437–446.

Lin, S. (1965). "Computer solutions of the traveling salesman problem". In: *Bell System Technical Journal* 44.10, pp. 2245–2269.

Lin, S. W., K. C. Ying, and C. Y. Huang (2013). "Minimising makespan in distributed permutation flowshops using a modified iterated greedy algorithm". In: *International Journal of Production Research* 51.16, pp. 5029–5038.

Lin, X., Z. Yang, and Y. Song (2011). "Intelligent stock trading system based on improved technical analysis and echo state network". In: *Expert systems with Applications* 38.9, pp. 11347–11354.

Liu, Q., S. Ullah, and C. Zhang (2011). "An improved genetic algorithm for robust permutation flowshop scheduling". In: *The International Journal of Advanced Manufacturing Technology* 56.1-4, pp. 345–354.

Liu, W.-Y., C.-C. Lin, C.-R. Chiu, Y.-S. Tsao, and Q. Wang (2014). "Minimizing the carbon footprint for the time-dependent heterogeneous-fleet vehicle routing problem with alternative paths". In: *Sustainability* 6.7, pp. 4658–4684.

Louis, S. J. and J. McDonnell (2004). "Learning with case-injected genetic algorithms". In: *Trans. Evol. Comp* 8.4, pp. 316–328.

Louis, Sushil J (2003). "Genetic learning from experience". In: *IEEE Congress on Evolutionary Computation*. Vol. 3, pp. 2118–2125.

Lourenço, H.R., O.C. Martin, and T. Stützle (2010). "Iterated local search: framework and applications". In: *Handbook of Metaheuristics*. Ed. by M. Gendreau and J.-Y. Potvin. Vol. 146. International Series in Operations Research & Management Science. Springer US, pp. 363–397.

Lu, Y., N. Zeng, X. Liu, and S. Yi (2015). "A new hybrid algorithm for bankruptcy prediction using switching particle swarm optimization and support vector machines". In: *Discrete Dynamics in Nature and Society* 2015.

Lwin, K., R. Qu, and G. Kendall (2014). "A learning-guided multi-objective evolutionary algorithm for constrained portfolio optimization". In: *Applied Soft Computing* 24, pp. 757–772.

Malakahmad, A., P. Md. Bakri, M. R. Md. Mokhtar, and N. Khalil (2014). "Solid waste collection routes optimization via GIS techniques in Ipoh City, Malaysia". In: *Procedia Engineering* Vol. 77, pp. 20–27.

Malkiel, B. G. (2003). "Passive investment strategies and efficient markets". In: *European Financial Management* 9.1, pp. 1–10.

Maniezzo, V., T. Stützle, and S. Vo (2009). *Matheuristics: Hybridizing metaheuristics and mathematical programming*. 1st. Springer Publishing Company, Incorporated.

Mansini, R., W. Ogryczak, and M. G. Speranza (2014). "Twenty years of linear programming based portfolio optimization". In: *European Journal of Operational Research* 234.2, pp. 518–535.

Marim, L. R., M. R. Lemes, and A. D. Pino (2003). "Neural-network-assisted genetic algorithm applied to silicon clusters". In: *Phys. Rev. A* 67.3.

Marinaki, M., Y. Marinakis, and C. Zopounidis (2010). "Honey bees mating optimization algorithm for financial classification problems". In: *Applied Soft Computing* 10.3, pp. 806–812.

Marinakis, Y., M. Marinaki, and N. Matsatsinis (2008). "A stochastic nature inspired metaheuristic for clustering analysis". In: *International Journal of Business Intelligence and Data Mining* 3.1, pp. 30–44.

Maringer, D. (2005). *Portfolio management with heuristic optimization*. Berlin: Springer.

Maringer, D. and H. Kellerer (2003). "Optimization of cardinality constrained portfolios with a hybrid local search algorithm". In: *OR Spectrum* 25.4, pp. 481–495.

Maringer, D. and O. Oyewumi (2007). "Index tracking with constrained portfolios". In: *Intelligent Systems in Accounting, Finance and Management* 15.1-2, pp. 57–71.

Markov, I., S. Varone, and M. Bierlaire (2016). "Integrating a heterogeneous fixed fleet and a flexible assignment of destination depots in the waste collection VRP with intermediate facilities". In: *Transportation Research Part B: Methodological* 84, pp. 256–273.

Martí, R., M. G. C. Resende, and C. Ribeiro (2013). "Multi-start methods for combinatorial optimization". In: *European Journal of Operational Research* 226.1, pp. 1–8.

Martin, O., S. W. Otto, and E. W. Felten (1992). "Large-step Markov chains for the TSP incorporating local search heuristics". In: *Oper. Res. Lett.* 11.4, pp. 219–224.

Martin, S., D. Ouelhadj, P. Beullens, E. Ozcan, A. A. Juan, and E.K. Burke (2016). "A Multi-Agent Based Cooperative Approach To Scheduling and Routing". In: *Eur. J. Oper. Res.* 254.1, pp. 169–178.

Mazza, D., A. Pagès-Bernaus, D. Tarchi, A. A. Juan, and G. E. Corazza (2016). "Supporting mobile cloud computing in smart cities via randomized algorithms". In: *IEEE Systems Journal*.

McQueen, G. and V. V. Roley (1993). "Stock prices, news, and business conditions". In: *Review of Financial Studies* 6.3, pp. 683–707.

Mendes, L., P. Godinho, and J. Dias (2012). "A Forex trading system based on a genetic algorithm". In: *Journal of Heuristics* 18.4, pp. 627–656.

Metaxiotis, K. and K. Liagkouras (2012). "Multiobjective evolutionary algorithms for portfolio management: a comprehensive literature review". In: *Expert Systems with Applications* 39.14, pp. 11685–11698.

Metropolis, N., A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller (1953). "Equation of state calculations by fast computing machines". In: *Journal of Chemical Physics* 21, pp. 1087–1092.

Michalski, R. S. (2000). "Learnable evolution model: Evolutionary processes guided by machine learning". In: *Mach. Learn.* 38.1-2, pp. 9–40.

Min, S.-H., J. Lee, and I. Han (2006). "Hybrid genetic algorithms and support vector machines for bankruptcy prediction". In: *Expert systems with applications* 31.3, pp. 652–660.

Mirabi, M., S.M.T. Fatemi-Ghomi, and F. Jolai (2010). "Efficient stochastic hybrid heuristics for the multi-depot vehicle routing problem". In: *Robotics and Computer-Integrated Manufacturing* 26.6, pp. 564–569.

Mishra, S. K., G. Panda, and R. Majhi (2014). "Constrained portfolio asset selection using multiobjective bacteria foraging optimization". In: *Operational Research* 14.1, pp. 113–145.

Mladenovic, N. (1995). "A variable neighborhood algorithm: A new metaheuristic for combinatorial optimization". In: *Abstracts of papers presented at Optimization Days*, p. 112.

Mladenović, N. and P. Hansen (1997). "Variable neighborhood search". In: *Computers & Operations Research* 24.11, pp. 1097–1100.

Moghaddam, B. F., R. Ruiz, and S. J. Sadjadi (2012). "Vehicle routing problem with uncertain demands: An advanced particle swarm algorithm". In: *Computers & Industrial Engineering* 62, pp. 306–317.

Molina, J., M. Laguna, Rafael. Martí, and R. Caballero (2007). "SSPMO: A scatter tabu search procedure for non-linear multiobjective optimization". In: *INFORMS Journal on Computing* 19.1, pp. 91–100.

Montero, E., M.-C. Riff, and B. Neveu (2014). "A beginner's guide to tuning methods". In: *Applied Soft Computing* 17, pp. 39–51.

Montoya-Torres, J., J. Lopez, S. Nieto, H. Felizzola, and N. Herazo-Padilla (2015). "A literature review on the vehicle routing problem with multiple depots". In: *Computers & Industrial Engineering* 79, pp. 115–129.

Moon, C., J. Kim, and S. Hur (2002). "Integrated process planning and scheduling with minimizing total tardiness in multi-plants supply chain". In: *Computers & Industrial Engineering* 43.1, pp. 331–349.

Moral-Escudero, R., R. Ruiz-Torrubiano, and A. Suárez (2006). "Selection of optimal investment portfolios with cardinality constraints". In: *2006 IEEE International Conference on Evolutionary Computation*. IEEE, pp. 2382–2388.

Moreno-Paredes, J. C., C. Mues, and L. C. Thomas (2013). "Credit scoring and credit control XIII Conference Proceedings". In: chap. A multi-objective decision framework for credit portfolio management.

Moreno-Vega, J. M. and B. Melián (2008). "Introduction to the special issue on variable neighborhood search". In: *Journal of Heuristics* 14.5, p. 403.

Muth, J. F. and G. L. Thompson (1963). *Industrial scheduling*. Prentice-Hall.

Myszkowski, P. B. and A. Bicz (2010). "Evolutionary algorithm in Forex trade strategy generation". In: *Computer Science and Information Technology (IMCSIT), Proceedings of the 2010 International Multiconference on*, pp. 81–88.

Naderi, B. and R. Ruiz (2010). "The distributed permutation flowshop scheduling problem". In: *Computers & Operations Research* 37.4, pp. 754–768.

— (2014). "A scatter search algorithm for the distributed permutation flowshop scheduling problem". In: *European Journal of Operational Research* 239.2, pp. 323–334.

Nag, B., B. Golden, and A. Assad (1988). "Vehicle routing: methods and studies". In: Amsterdam: Elsevier. Chap. Vehicle routing with site dependencies, pp. 149–159.

Nance, R. E. and R. G. Sargent (2002). "Perspectives on the evolution of simulation". In: *Operations Research* 50.1, pp. 161–172.

Nawaz, M., J. Enscore, and I. Ham (1983). "A heuristic algorithm for the m-machine, n-job flowshop sequencing problem". In: *Omega* 11, 91—95.

Neirotti, P., A. De Marco, A. C. Cagliano, G. Mangano, and F. Scorrano (2014). "Current trends in smart city initiatives: some stylised facts". In: *Cities* 38, pp. 25–36.

Ni, H. and Y. Wang (2013). "Stock index tracking by Pareto efficient genetic algorithm". In: *Applied Soft Computing* 13.12, pp. 4519–4535.

Niknamfar, A. H. and S. T. A. Niaki (2016). "Fair profit contract for a carrier collaboration framework in a green hub network under soft time-windows: Dual lexicographic max–min approach". In: *Transportation Research Part E: Logistics and Transportation Review* 91, pp. 129–151.

Nikolaev, A. G. and S. H. Jacobson (2010). "Simulated annealing". In: *Handbook of metaheuristics*. Springer, pp. 1–39.

Nolz, P., N. Absi, and D. Feillet (2014). "A stochastic inventory routing problem for infectious medical waste collection". In: *Networks* 63, pp. 82–95.

Nunez-Letamendia, L. (2007). "Fitting the control parameters of a genetic algorithm: An application to technical trading systems design". In: *European journal of operational research* 179.3, pp. 847–868.

Nuortio, T., J. Kytöjoki, H. Niska, and O. Bräysy (2006). "Improved route planning and scheduling of waste collection and transport". In: *Expert Systems with Applications* 30, pp. 223–232.

Ombuki-Berman, B. M., A. Runka, and F. T. Hanshar (2007). "Waste collection vehicle routing problem with time windows using multi-objective genetic algorithms". In: *Proceedings of the third IASTED International Conference on Computational Intelligence*. Anaheim, US, pp. 91–97.

Oreski, S. and G. Oreski (2014). "Genetic algorithm-based heuristic for feature selection in credit risk assessment". In: *Expert systems with applications* 41.4, pp. 2052–2064.

Oreski, S., D. Oreski, and G. Oreski (2012). "Hybrid system with genetic algorithm and artificial neural networks and its application to retail credit risk assessment". In: *Expert systems with applications* 39.16, pp. 12605–12617.

Pathak, B. K., S. Srivastava, and K. Srivastava (2008). "Neural network embedded multiobjective genetic algorithm to solve non-linear time-cost tradeoff problems of project scheduling". In: *Journal of scientific and industrial research* 67.2, pp. 124–131.

Pavón, R., F. Díaz, R. Laza, and V. Luzón (2009). "Automatic parameter tuning with a Bayesian case-based reasoning system. A case of study". In: *Expert Systems With Applications* 36.2, pp. 3407–3420.

Pelikan, M., D. E. Goldberg, and F. G. Lobo (2002). "A survey of optimization by building and using probabilistic models". In: *Comput. Optim. Appl.* 21.1, pp. 5–20.

Pereira, I., A. Madureira, P. B. M. Oliveira, and A. Abraham (2013). "Tuning meta-heuristics using multi-agent learning in a scheduling system". In: *Transactions on Computational Science XXI*. Springer Berlin Heidelberg, pp. 190–210.

Pisinger, D. and S. Ropke (2007). "A general heuristic for vehicle routing problems". In: *Computers and Operations Research* 34.8, pp. 2403–2435.

Polya, G. (1945). *How to solve it: a new aspect of mathematical model*. Tech. rep. Princeton University Press Princeton.

Pongcharoen, P., W. Chainate, and P. Thapatsuwan (2007). "Exploration of genetic parameters and operators through travelling salesman problem". In: *Science Asia* 33.2, pp. 215–222.

Prasanna, S. and D. Ezhilmaran (2015). "Stock market prediction using clustering with meta-heuristic approaches". In: *Gazi University Journal of Science* 28.3, pp. 395–403.

Pukthuanthong, K. and R. Roll (2011). "Gold and the Dollar (and the Euro, Pound, and Yen)". In: *Journal of Banking and Finance* 35.8, pp. 2070–2083.

Quintero-Araujo, C. L., J. P. Caballero-Villalobos, A. A. Juan, and J. R. Montoya-Torres (2016). "A biased-randomized metaheuristic for the capacitated location routing problem". In: *International Transactions in Operational Research*.

Rabbani, M., M. A. Bajestani, and G. B. Khoshkhou (2010). "A multi-objective particle swarm optimization for project selection problem". In: *Expert Systems with Applications* 37.1, pp. 315–321.

Rachev, S. T., B. Racheva-Iotova, S. V. Stoyanov, and F. J. Fabozzi (2010). "Risk management and portfolio optimization for volatile markets". In: *Handbook of Portfolio Construction*, pp. 493–508.

Raft, O.M. (1982). "A modular algorithm for an extended vehicle scheduling problem". In: *European Journal of Operational Research* 11, pp. 67–76.

Ramos, I. C. O., M. C. Goldbarg, E. G. Goldbarg, and A. D. D. Neto (2005). "Logistic regression for parameter tuning on an evolutionary algorithm". In: *Evolutionary Computation, 2005. The 2005 IEEE Congress on*. Vol. 2. IEEE, pp. 1061–1068.

Ramos, T. R. P., M. I. Gomes, and A. P. B.arbosa-Póvoa (2014). "Economic and environmental concerns in planning recyclable waste collection systems". In: *Transportation Research Part E: Logistics and Transportation Review* 62, pp. 34–54.

Ramsey, C. L. and J. J. Grefenstette (1993). "Case-based initialization of genetic algorithms". In: *Proceedings of the 5th International Conference on Genetic Algorithms*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., pp. 84–91.

Rasheed, K. and H. Hirsh (2000). "Informed operators: speeding up genetic-algorithm-based design optimization using reduced models". In: *Proceedings of the 2nd Annual Conference on Genetic and Evolutionary Computation*. Morgan Kaufmann Publishers Inc., pp. 628–635.

Reboredo, J. C. (2013). "Is gold a safe haven or a hedge for the US dollar? Implications for risk management". In: *Journal of Banking & Finance* 37.8, pp. 2665–2676.

Regis, R. G. (2014). "Evolutionary programming for high-dimensional constrained expensive black-box optimization using radial basis functions". In: *IEEE Transactions on Evolutionary Computation* 18.3, pp. 326–347.

Reid, S. G. and K. M. Malan (2015). "Constraint handling methods for portfolio optimization using particle swarm optimization". In: *Computational Intelligence, 2015 IEEE Symposium Series on*, pp. 1766–1773.

Renaud, J., G. Laporte, and F.F. Boctor (1996). "A tabu search heuristic for the multi-depot vehicle routing problem". In: *Computers & Operations Research* 23.3, pp. 229–235.

Reyes, L., L. Calvet, A. A. Juan, and J. Faulin (submitted). "Sustainable urban freight transport considering multiple capacitated depots". In: *A*.

Ribas, I., R. Companys, and X. Tort-Martorell (2017). "Efficient heuristics for the parallel blocking flow shop scheduling problem". In: *Expert Systems with Applications* 74, pp. 41–54.

Ribeiro, M. H., A. Plastino, and S. L. Martins (2006). "Hybridization of GRASP metaheuristic with data mining techniques". In: *Journal of Mathematical Modelling and Algorithms* 5.1, pp. 23–41.

Rice, J. R. (1976). "The algorithm selection problem". In: *Adv. Comput.* 15, pp. 65–118.

Ridge, E. and D. Kudenko (2007). "Analyzing heuristic performance with response surface models: prediction, optimization and robustness". In: *Proceedings of the 9th annual conference on Genetic and evolutionary computation*. ACM, pp. 150–157.

Ries, J. (2009). "Instance-based flexible parameter tuning for meta-heuristics using fuzzy-logic". PhD thesis. University of Portsmouth.

Ries, J., P. Beullens, and D. Salt (2012). "Instance-specific multi-objective parameter tuning based on fuzzy logic". In: *European Journal of Operational Research* 218.2, pp. 305–315.

Rodammer, F. A. and K. P. White (1988). "A recent survey of production scheduling". In: *IEEE Transactions on Systems, Man and Cybernetics* 18.6, pp. 841–851.

Rousseeuw, P. J. (1987). "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis". In: *Journal of computational and applied mathematics* 20, pp. 53–65.

Roy, B. (2010). "Robustness in operational research and decision aiding: a multi-faceted issue". In: *European Journal of Operational Research* 200.3, pp. 629–638.

Ruiz, R. and T. Stützle (2007). "A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem". In: *European Journal of Operational Research* 177.3, pp. 2033–2049.

Ruiz, X., L. Calvet, J. Ferrarons, and A. A. Juan (2015). "SmartMonkey: a web browser tool for solving combinatorial optimization problems in real time". In: *Proceedings of the 2015 Int. Conf. of the Forum for Interdisciplinary Mathematics*. Barcelona, Spain.

Ruiz-Torrubiano, R. and A. Suárez (2009). "A hybrid optimization approach to index tracking". In: *Annals of Operations Research* 166.1, pp. 57–71.

Salhi, S. and M. Sari (1997). "A multi-level composite heuristic for the multi-depot vehicle fleet mix problem". In: *European Journal of Operational Research* 103, pp. 95–112.

Santos, H. G., L. S. Ochi, E. H. Marinho, and L. M. A. Drummond (2006). "Combining an evolutionary algorithm with data mining to solve a single-vehicle routing problem". In: *Neurocomputing* 70.1, pp. 70–77.

Santos, L. F., M. H. Ribeiro, A. Plastino, and S. L. Martins (2005). "A hybrid GRASP with data mining for the maximum diversity problem". In: *International Workshop on Hybrid Metaheuristics*. Springer, pp. 116–127.

Santos, L. F., S. L. Martins, and A. Plastino (2008). "Applications of the DM-GRASP heuristic: a survey". In: *International Transactions in Operational Research* 15.4, pp. 387–416.

Schaerf, A. (2002). "Local search techniques for constrained portfolio selection problems". In: *Computational Economics* 20.3, pp. 177–190.

Scozzari, A., F. Tardella, S. Paterlini, and T. Krink (2013). "Exact and heuristic approaches for the index tracking problem with UCITS constraints". In: *Annals of Operations Research* 205.1, pp. 235–250.

Seidgar, H., M. Kiani, M. Abedi, and H. Fazlollahtabar (2014). "An efficient imperialist competitive algorithm for scheduling in the two-stage assembly flow shop problem". In: *International Journal of Production Research* 52.4, pp. 1240–1256.

Senjyu, T., A. Y. Saber, T. Miyagi, K. Shimabukuro, N. Urasaki, and T. Funabashi (2005). "Fast technique for unit commitment by genetic algorithm based on unit clustering". In: *IEE Proceedings-Generation, Transmission and Distribution* 152.5, pp. 705–713.

Shelokar, P. S., V. K. Jayaraman, and B. D. Kulkarni (2004). "An ant colony approach for clustering". In: *Anal. Chim. Acta* 509.2, pp. 187–195.

Shin, K.-S. and Y.-J. Lee (2002). "A genetic algorithm application in bankruptcy prediction modeling". In: *Expert Systems with Applications* 23.3, pp. 321–328.

Silvennoinen, A. and S. Thorp (2013). "Financialization, crisis and commodity correlation dynamics". In: *Journal of International Financial Markets, Institutions and Money* 24, pp. 42–65.

Sluga, A., P. Butala, and G. Bervar (1998). "A multi-agent approach to process planning and fabrication in distributed manufacturing". In: *Computers & Industrial Engineering* 35.3, pp. 455–458.

Smith, J. E. (2008). "Self-adaptation in evolutionary algorithms for combinatorial optimisation". In: *Adaptive and Multilevel Metaheuristics*. Springer, pp. 31–57.

Smith-Miles, K., D. Baatar, B. Wreford, and R. Lewis (2014). "Towards objective measures of algorithm performance across instance space". In: *Comput. Oper. Res.* 45, pp. 12–24.

Smith-Miles, K. A. (2009). "Cross-disciplinary perspectives on meta-learning for algorithm selection". In: *ACM Computing Surveys* 41.1, p. 6.

Soleimani, H., H. R. Golmakani, and M. H. Salimi (2009). "Markowitz-based portfolio selection with minimum transaction lots, cardinality constraints and regarding sector capitalization using genetic algorithm". In: *Expert Systems with Applications* 36.3, pp. 5058–5063.

Solomon, M. M. (1987). "Algorithms for the vehicle routing and scheduling problems with time windows". In: *Operations Research* Vol. 35, pp. 254–265.

Son, L. H. (2014). "Optimizing municipal solid waste collection using chaotic particle swarm optimization in GIS based environments: A case study at Danang city, Vietnam". In: *Expert Systems with Applications* 41, pp. 8062–8074.

Sörensen, K. (2015). "Metaheuristics—the metaphor exposed". In: *International Transactions in Operational Research* 22.1, pp. 3–18.

Stanley, K. O. and R. Miikkulainen (2002). "Evolving neural networks through augmenting topologies". In: *Evol. Comput.* 10.2, pp. 99–127.

Stewart, Jr. W. R. and B.L. Golden (1983). "Stochastic vehicle routing: A comprehensive approach". In: *European Journal of Operational Research* 14, pp. 371–385.

Streichert, F., H. Ulmer, and A. Zell (2003). "Evolutionary algorithms and the cardinality constrained portfolio selection problem". In: *Operations Research Proceedings 2003, Selected Papers of the International Conference on Operations Research (OR 2003)*.

Stummer, C. and M. Sun (2005). "New multiobjective metaheuristic solution procedures for capital investment planning". In: *Journal of Heuristics* 11.3, pp. 183–199.

Suman, B. and P. Kumar (2006). "A survey of simulated annealing as a tool for single and multiobjective optimization". In: *The Journal of the Operational Research Society* 10.57, pp. 1143–1160.

Sung, C. S. and J. Juhn (2009). "Makespan minimization for a 2-stage assembly scheduling problem subject to component available time constraint". In: *International Journal of Production Economics* 119.2, pp. 392–401.

Sung, C. S. and H. A. Kim (2008). "A two-stage multiple-machine assembly scheduling problem for minimizing sum of completion times". In: *International Journal of Production Economics* 113.2, pp. 1038–1048.

Surekha, P. and S. Sumathi (2011). "Solution to MDVRP using genetic algorithms". In: *World Applied Programming* 1, pp. 118–131.

Taillard, E. (1990). "Some efficient heuristic methods for the flow shop sequencing problem". In: *European Journal of Operational Research* 47.1, pp. 65–74.

— (1993). "Benchmarks for basic scheduling problems". In: *European Journal of Operational Research* 64, 278—285.

Talbi, E.-G. (2006). *Parallel combinatorial optimization*. Vol. 58. John Wiley & Sons.

— (2009). *Metaheuristics: From design to implementation*. New Jersey: John Wiley & Sons.

— (2013). "Combining metaheuristics with mathematical programming, constraint programming and machine learning". In: *4OR* 11.2, pp. 101–150.

Tan, K. C., C. Y. Cheong, and C. K. Goh (2007). "Solving multiobjective vehicle routing problem with stochastic demand via evolutionary computation". In: *Discrete optimization* 177, pp. 813–839.

Tatarakis, A. and I. Minis (2009). "Stochastic single vehicle routing with a predefined customer sequence and multiple depot returns". In: *European Journal of Operational Research* 197, pp. 557–571.

Tauhid, S., Z. Jannat, and F. Mahmud (2012). "A novel three-phase approach for solving multi-depot vehicle routing problem with stochastic demand". In: *Algorithms Research* 1, pp. 15–19.

Tavares, G., Z. Zsigraiova, V. Semiao, and M. Carvalho (2009). "Optimisation of MSW collection routes for minimum fuel consumption using 3D GIS modelling". In: *Waste Management* 29, pp. 1176–1185.

Team, R Development Core (2008). *R: a language and environment for statistical computing*. R Foundation for Statistical Computing.

Teixeira, J., A. P. Antunes, and J. P. de Sousa (2004). "Recyclable waste collection planning-a case study". In: *European Journal of Operational Research* 158, pp. 543–554.

Teixeira, L. A. and A. L. I. De Oliveira (2010). "A method for automatic stock trading combining technical analysis and nearest neighbor classification". In: *Expert systems with applications* 37.10, pp. 6885–6890.

Tenne, Y. and C.-K. Goh (2010). *Computational intelligence in expensive optimization problems*. Vol. 2. Springer Science & Business Media.

Thangiah, S. R. and S. Salhi (2001). "Genetic clustering: an adaptive heuristic for the multi-depot vehicle routing problem". In: *Applied Artificial Intelligence* 15.4, pp. 361–383.

Theodoridis, S. and K. Koutroumbas (2009). *Pattern recognition*. Vol. 74. John Wiley & Sons.

Thomaidis, N. S. (2011). "A soft computing approach to enhanced indexation". In: *Natural Computing in Computational Finance*. Springer, pp. 61–77.

Tillman, F. A. (1969). "The multiple terminal delivery problem with probabilistic demands". In: *Transportation Science* 3, pp. 192–204.

Tillman, F. A. and T. M. Cain (1972). "An upper bound algorithm for the single and multiple terminal delivery problem". In: *Management Science* 18.11, pp. 664–682.

Tuba, M. and N. Bacanin (2014a). "Artificial bee colony algorithm hybridized with firefly algorithm for cardinality constrained mean-variance portfolio selection problem". In: *Appl. Math. Inf. Sci* 8.6, pp. 2831–2844.

— (2014b). "Upgraded firefly algorithm for portfolio optimization problem". In: *Computer Modelling and Simulation (UKSim), 2014 UKSim-AMSS 16th International Conference on*. IEEE, pp. 113–118.

Tyasnurita, R., E. Ozcan, S. Asta, and R. John (2015). "Improving Performance of a Hyper-heuristic Using a Multilayer Perceptron for Vehicle Routing". In: *15th Annual Workshop on Computational Intelligence*. Springer. Lancaster, UK.

Ubeda, S., F.J. Arcelus, and J. Faulin (2011). "Green logistics at Eroski: A case study". In: *International Journal of Production Economics* 131.1, pp. 44–51.

Urli, B. and F. Terrien (2010). "Project portfolio selection model, a realistic approach". In: *International Transactions in Operational Research* 17.6, pp. 809–826.

Vairaktarakis, G. and M. Elhafsi (2000). "The use of flowlines to simplify routing complexity in two-stage flowshops". In: *IIE Transactions (Institute of Industrial Engineers)* 32.8, pp. 687–699.

Viana, A., J. P. Sousa, and M. A. Matos (2005). "Constraint oriented neighbourhoods – A new search strategy in metaheuristics". In: *Metaheuristics: progress as real problem solvers*. Springer, pp. 389–414.

Vidal, T., T G Crainic, M. Gendreau, N. Lahrichi, and W. Rei (2012). "A hybrid genetic algorithm for multi-depot and periodic vehicle routing problems". In: *Operations Research* 60, pp. 611–624.

Wang, B. (1997). *Integrated product, process and enterprise design*. London. ISBN: 0412620200.

Wang, F., L. H. Philip, and D. W. Cheung (2014a). "Combining technical trading rules using particle swarm optimization". In: *Expert Systems with Applications* 41.6, pp. 3016–3026.

Wang, J., K. Guo, and S. Wang (2010). "Rough set and tabu search based feature selection for credit scoring". In: *Procedia Computer Science* 1.1, pp. 2425–2432.

Wang, J., A.-R. Hedar, S. Wang, and J. Ma (2012). "Rough set and scatter search metaheuristic based feature selection for credit scoring". In: *Expert Systems with Applications* 39.6, pp. 6123–6128.

Wang, J., K. Tang, J. Lozano, and X. Yao (2015). "Estimation of Distribution Algorithm with Stochastic Local Search for Uncertain Capacitated Arc Routing Problems". In: *IEEE T. Evolut. Comput.* 20.c, pp. 1–1.

Wang, L., W. Shen, and Q. Hao (2006). "An overview of distributed process planning and its integration with scheduling". In: *International Journal of Computer Applications in Technology* 26.1/2, p. 3.

Wang, X., H. Kopfer, and M. Gendreau (2014b). "Operational transportation planning of freight forwarding companies in horizontal coalitions". In: *European Journal of Operational Research* 237.3, pp. 1133–1141.

Wang, X., T. Fan, W. Li, R. Yu, D. Bullock, B. Wu, and P. Tremont (2016). "Speed variation during peak and off-peak hours on urban arterials in Shanghai". In: *Transportation Research Part C: Emerging Technologies* 67, pp. 84–94.

Wang, Y. and I.H. Witten (1996). *Induction of model trees for predicting continuous classes*. Tech. rep. Working paper 96/23. Hamilton, New Zealand: University of Waikato.

Woodside-Oriakhi, M., C. Lucas, and J. E. Beasley (2011). "Heuristic algorithms for the cardinality constrained efficient frontier". In: *European Journal of Operational Research* 213.3, pp. 538–550.

Wu, T.-H., C. Low, and J.-W. Bai (2002). "Heuristic solutions to multi-depot location-routing problems". In: *Computers & Operations Research* 29.10, pp. 1393–1415.

Xiao, Y. and A. Konak (2015). "Green vehicle routing problem with time-varying traffic congestion". In: *Proceedings of the 14th INFORMS Computing Society Conference*, pp. 134–148.

Xu, J., S. Y. Chiu, and F. Glover (1998). "Fine-tuning a tabu search algorithm with statistical tests". In: *International Transactions in Operational Research* 5.3, pp. 233–244.

Yalcinoz, T. and H. Altun (2001). "Power economic dispatch using a hybrid genetic algorithm". In: *IEEE Power Engineering Review* 21.3, pp. 59–60.

Yang, S., Q. H. Liu, J. Lu, S. L. Ho, G. Ni, P. Ni, and S. Xiong (2009). "Application of support vector machines to accelerate the solution speed of metaheuristic algorithms". In: *IEEE T. Magn.* 45.3, pp. 1502–1505.

Yang, W. H., K. Mathur, and R. H. Ballou (2000). "Stochastic vehicle routing problem with restocking". In: *Transportation Science* 34, pp. 99–112.

Yao, X. (1999). "Evolving artificial neural networks". In: *Proceedings of the IEEE* 87.9, pp. 1423–1447.

Yoo, S.-H. and S.-B. Cho (2004). "Partially evaluated genetic algorithm based on fuzzy c-means algorithm". In: *International Conference on Parallel Problem Solving from Nature*. Springer, pp. 440–449.

You, L. and R. T. Daigler (2012). "A Markowitz optimization of commodity futures portfolios". In: *The Journal of Futures Markets* 33.4, pp. 343–368.

Yusuf, I. (2014). "Solving multi-depot, heterogeneous, site dependent and asymmetric VRP using three steps heuristic". In: *Journal of Algorithms and Optimization* 2.2, pp. 28–42.

Zennaki, M. and A. Ech-Cherif (2010). "A new machine learning based approach for tuning metaheuristics for the solution of hard combinatorial Optimization problems". In: *Journal of Applied Sciences* 10, pp. 1991–2000.

Zhang, H. and R. Ren (2010). "High frequency foreign exchange trading strategies based on genetic algorithms". In: *Networks Security Wireless Communications and Trusted Computing (NSWCTC), 2010 Second International Conference on*. Vol. 2, pp. 426–429.

Zhang, J., Z.-H. Zhang, Y. Lin, N. Chen, Y.-J. Gong, J.-H. Zhong, H. Chung, Y. Li, and Y.-H. Shi (2011). "Evolutionary Computation Meets Machine Learning: A Survey". In: *Comp. Intell. Mag.* 6.4, pp. 68–75.

Zhang, J., Y. Zhao, W. Xue, and J. Li (2015). "Vehicle routing problem with fuel consumption and carbon emission". In: *International Journal of Production Economics* 170, pp. 234–242.

Zhang, S., C. K. M. Lee, K. Wu, and K. L. Choy (2016). "Multi-objective optimization for sustainable supply chain network design considering multiple distribution channels". In: *Expert Systems with Applications* 65, pp. 87–99.

Zhang, X. and S. Van De Velde (2012). "Approximation algorithms for the parallel flow shop problem". In: *European Journal of Operational Research* 216.3, pp. 544–552.

Zhou, A. and Q. Zhang (2010). "A surrogate-assisted evolutionary algorithm for minimax optimization". In: *IEEE Congress on Evolutionary Computation*. IEEE, pp. 1–7.

Zhou, Q. and X. Cui (2008). "Research on multiobjective flow shop scheduling with stochastic processing times and machine breakdowns". In: *IEEE International Conference on Service Operations and Logistics, and Informatics*. Vol. 2. IEEE, pp. 1718–1724.

Zhou, Z., Y. S. Ong, M. H. Nguyen, and D. Lim (2005). "A study on polynomial regression and gaussian process global surrogate model in hierarchical surrogate-assisted evolutionary algorithm". In: *IEEE Congress on Evolutionary Computation*. Vol. 3. IEEE, pp. 2832–2839.

Zhu, H., Y. Wang, K. Wang, and Y. Chen (2011). "Particle swarm optimization for the constrained portfolio optimization problem". In: *Expert Systems with Applications* 38.8, pp. 10161–10169.

Part IV

APPENDICES

Appendix A

Journal papers indexed in ISI JCR

A.1 Learnheuristics: Hybridizing metaheuristics with machine learning for optimization with dynamic inputs

Laura Calvet, Jésica de Armas, David Masip, Angel A. Juan

Dept. of Computer Science – IN3, Open University of Catalonia, Castelldefels, Spain
e-mail: {lcalvetl, jde_armasa, ajuanp, dmasipr}@uoc.edu

Abstract

This paper reviews the existing literature on the combination of metaheuristics with machine learning methods and then introduces the concept of learnheuristics, a novel type of hybrid algorithms. Learnheuristics can be used to solve combinatorial optimization problems with dynamic inputs (COPDIs). In these COPDIs, the problem inputs (elements either located in the objective function or in the constraints set) are not fixed in advance as usual. On the contrary, they might vary in a predictable (non-random) way as the solution is partially built according to some heuristic-based iterative process. For instance, a consumer's willingness to spend on a specific product might change as the availability of this product decreases and its price rises. Thus, these inputs might take different values depending on the current solution configuration. These variations in the inputs might require from a coordination between the learning mechanism and the metaheuristic algorithm: at each iteration, the learning method updates the inputs model used by the metaheuristic.

Keywords: Hybrid Algorithms, Combinatorial Optimization, Metaheuristics, Machine Learning, Dynamic Inputs.

1. Introduction

Operations Research (OR) is a well-established field with a huge and active research community. One of its main goals is to support decision-making processes in complex scenarios, i.e., providing optimal (or near-optimal) solutions to combinatorial optimization problems (COPs) defined by a given objective function and a set of realistic constraints. The number of applications is immense, e.g.: transportation and logistics, finance, production, telecommunication systems, etc. A noticeable part of the efforts developed by the OR community has focused on developing exact methods to find optimal solutions to a wide range of COPs. When dealing with NP-hard COPs, this usually requires simplifying somewhat the model and/or addressing only small- and medium-sized instances to avoid incurring in prohibitive computing times. Another noticeable part of the efforts has been invested in developing heuristic and metaheuristic approaches that cannot guarantee optimality of the provided solutions but are usually more powerful in terms of the size of the instances they can solve in reasonable computing times (Talbi, 2009). Additionally, these approximated methods are quite flexible, which makes them suitable for tackling more realistic and rich models. While heuristics are simple and fast procedures based on the specific COP being addressed, metaheuristics are general templates that can be easily adapted to a huge variety of COPs.

The OR community shows a growing interest in coping with increasingly challenging COPs, such as stochastic COPs (in which some of the problem inputs are random variables) and dynamic COPs (in which some of the problem inputs evolve over time). This might be due to several

factors, including: *(i)* the rich characteristics of real-life problems frequently faced by modern companies in sectors such as logistics and transportation (Caceres et al., 2014); *(ii)* the technological development; *(iii)* the availability of vast amounts of Internet-based data; and *(iv)* a shift to a more data-driven culture. During the last years, hybrid approaches have been extensively employed due to their success when dealing with realistic problems, among others: those combining different metaheuristics (Talbi, 2013), matheuristics (i.e., metaheuristics combined with mathematical programming) (Maniezzo et al., 2009), and simheuristics (i.e., metaheuristics combined with simulation) (Juan et al., 2015a).

The hybridization of metaheuristics with machine learning techniques is an emerging research field in the OR community. In this context, the main contributions of this paper are: *(i)* providing a survey on the existing works combining metaheuristics with machine learning techniques, as well as a classification of the most relevant ones; and *(ii)* proposing a novel 'learnheuristic' framework, combining a heuristic-based constructive procedure with machine learning, to deal with a special kind of COPs with dynamic inputs (COPDIs). In these COPDIs, the inputs are deterministic (i.e., non-stochastic) but, instead of being fixed in advance, they vary according to the structure of the solution (i.e., they change as the solution is being constructed following a heuristic-based iterative process). In this sense, these COPDIs represent an extension of the classical deterministic COPs in which all inputs are given in advance and are immutable. An example of such a COPDI is given next for illustrative purposes. Suppose there is a set of heterogeneous radio access technologies (RATs) that provide pay-per-use services to a group of users. Each user has to be assigned to just one RAT, and each RAT can serve only a limited number of users. Being a pay-per-use service, the goal here is to maximize the total benefit, which depends on the customers' demands. Several scenarios may be described based on the nature of the customers' demands (Figure A.1): *(i)* they are deterministic, static (do not change over time), and can be computed or accurately estimated; *(ii)* they contain some degree of uncertainty but can be modeled as random variables or using fuzzy techniques; and *(iii)* they are dynamic in the sense that they depend on the solution characteristics (e.g., the number of users connected to the same RAT, which has an effect on the service quality and, therefore, on the customers' demands of that service).

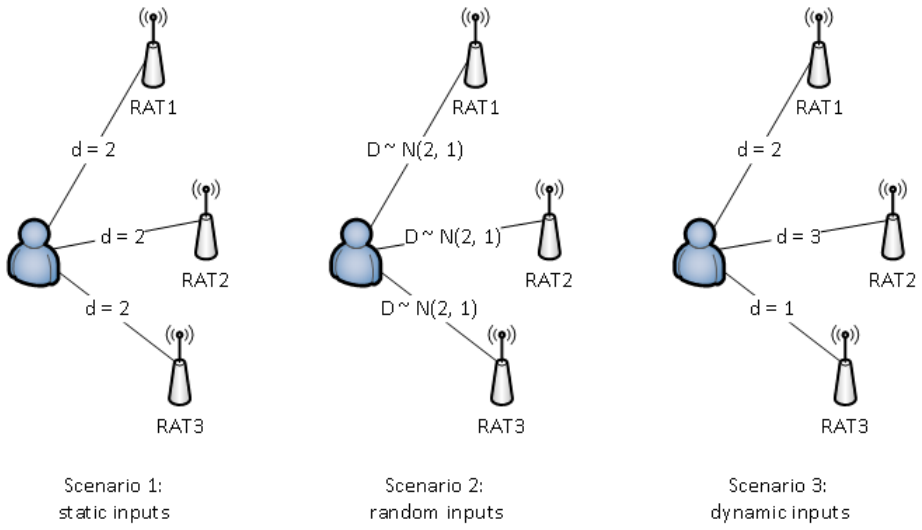


FIGURE A.1: Different scenarios according to the nature of the inputs.

While the first case corresponds to a classical deterministic COP, the second case introduces a level of uncertainty that usually requires the use of stochastic programming, simulation-optimization, or fuzzy methods. In this paper we focus on the third case, and propose the use of learnheuristic algorithms, in which the learning mechanism updates the input values as the solution is iteratively constructed using the heuristic logic (Calvet et al., 2016d). Notice, however, that not all the metaheuristics rely on constructive procedures to generate new solutions. Thus, for instance, evolutionary algorithms or scatter search algorithms typically generate new solutions by simply combining already existing solutions, which might have been generated at random.

The rest of the paper is structured as follows: Section 2 provides a brief introduction to metaheuristics and machine learning, and proposes a classification of hybrid works combining both

methodologies. Section 3 presents an overview of works in which machine learning techniques have been used to enhance the performance of metaheuristics, while Section 4 reviews publications in which metaheuristics have been used to improve machine learning methods. Section 5 provides a formal description of the COPDIs we aim to solve and explains the main ideas behind our learnheuristics solving framework. Section 6 discusses potential applications of learnheuristics in different fields. Section 7 provides a numerical experiment that illustrates the use of learnheuristics in a vehicle routing problem with dynamic demands. Finally, Section 8 summarizes the main conclusions and identifies some future research lines.

2. Metaheuristics and machine learning

Definitions and evolution of the number of works

Metaheuristics represent a heterogeneous family of algorithms designed to solve a high number of complex COPs without having to deeply adapt them to each problem. They do not guarantee optimal solutions, but may provide near-optimal ones in a reasonable amount of computing time. A number of them are nature-inspired, include stochastic components, and have several parameters that must be fine-tuned and may interact (Boussaïd et al., 2013). Figure A.2 includes some of the most popular metaheuristics (first works are cited): ant colony optimization (ACO) (Dorigo, 1992), artificial immune systems (AIS) (Farmer et al., 1986), genetic algorithms (GA) (Holland, 1962), greedy randomized adaptive search procedure (GRASP) (Feo and Resende, 1989), iterated local search (ILS) (Martin et al., 1992), particle swarm optimization (PSO) (Kennedy and Eberhart, 1995), scatter search (SS) (Glover, 1977), simulated annealing (SA) (Kirkpatrick, 1984), tabu search (TS) (Glover, 1986) and variable neighborhood search (VNS) (Mladenovic, 1995). They are grouped according to the following criteria: *(i)* single-solution versus population-based metaheuristics (SMs and PMs, respectively); *(ii)* whether they use memory; and *(iii)* whether they are nature-inspired. The success of the first implementations of metaheuristics aroused the interest of journals in new versions of these methods, which increased the number of authors exploring this topic. Unfortunately, some publications add only marginal contributions to the already existing frameworks (Sörensen, 2015). As stated in Feo and Resende (1995), the effectiveness of a given metaheuristic depends upon its ability to adapt to a particular instance problem, avoid entrapment at local optima, and exploit the structure of the problem. In addition, the authors highlight the potential benefit of restart procedures, controlled randomization, efficient data structures, and pre-processing. A few fields where they are commonly applied are: logistics and transportation, telecommunications, production and scheduling, bioinformatics, finance, smart cities, cryptology, and nutrition, among many others. The reader interested in a complete review of metaheuristics is referred to Gendreau and Potvin (2010).

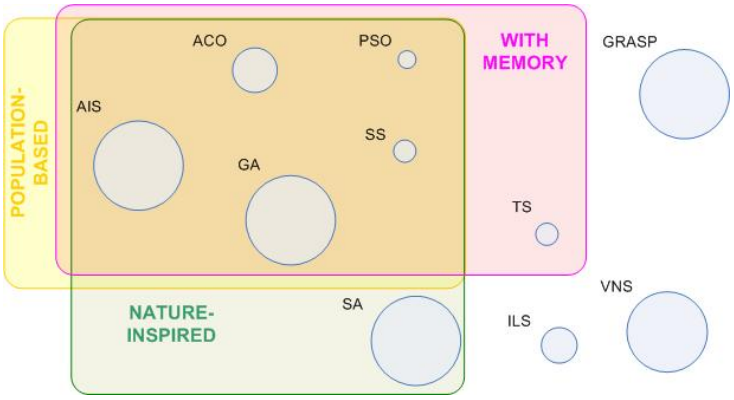


FIGURE A.2: Main metaheuristics grouped by different criteria. Circles' size is proportional to the number of Google Scholar indexed articles, from 2006 to 2015, that include the complete name of the specific metaheuristic and "metaheuristics" or "heuristics" in the article (March 15, 2016).

Machine learning is a subfield of computer science and artificial intelligence that encompasses a number of algorithms which learn from a dataset composed of examples or observations and are capable of making predictions (Barber, 2012; Lantz, 2013). There are three styles of learning:

supervised, unsupervised, and semi-supervised. The first relies on a set of procedures for function approximation. Based on a database of labelled samples, the goal is to predict a response variable (or output) from the explanatory variables (or inputs). Supervised methods are used in the following tasks: regression, classification, dimension reduction, time-series prediction, and reinforcement learning. In contrast, unsupervised learning does not include any response variable, and attempts to find compact descriptions of the data. The main tasks are: anomaly detection, dimension reduction, time-series modeling, and latent variable models. Finally, semi-supervised learning is similar to supervised learning but, in this case, not all the examples have associated an output value. Semi-supervised methods are very useful in problems where large amounts of unlabelled samples are available, and only a few of them can be manually labelled. Typical examples are visual object recognition, where milions of untagged images are publicly available, or natural language processing. The most popular applications of machine learning techniques include search engines, robotics, computer vision, finance, bioinformatics, finance, insurance, etc.

According to data from Google Scholar, both fields may be considered young (Figure A.3). Although the use of machine learning techniques is much more extended, metaheuristics are more employed in the context of COPs. An example of machine learning applied to solve these problems is the work developed in neural networks for solving COPs, mainly in vehicle routing problems (Potvin and Smith, 2003; Smith, 1999).

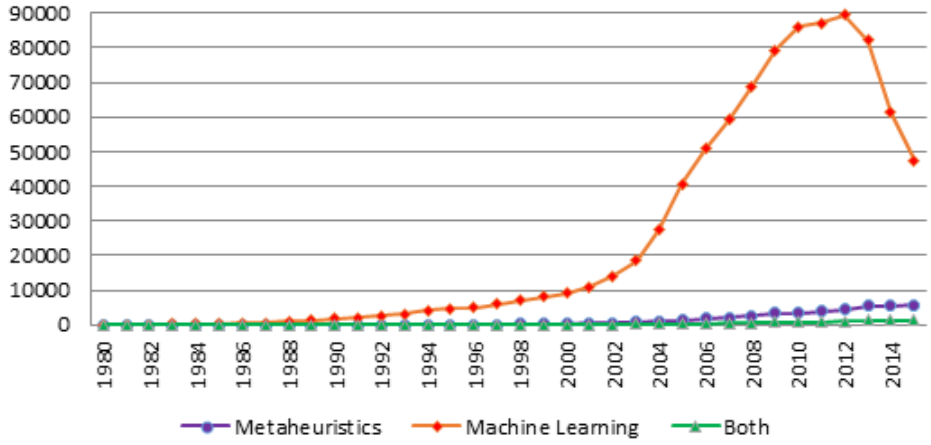


FIGURE A.3: Evolution of the number of works in Google Scholar (March 15, 2016). The number of works from the fields of metaheuristics and statistics or data mining were 1880 and 785 in 2015, respectively.

Reviews on the combination of metaheuristics and machine learning

The existing literature analyzing the hybridization of metaheuristics and machine learning may be mainly divided into two groups: works were machine learning is employed to enhance metaheuristics, and those in which metaheuristics are used to improve the performance of machine learning techniques.

Regarding the first group, there are several works providing overviews from different points of view. For instance, the emergence of hybrid metaheuristics is studied in Talbi (2013), which includes the combination of metaheuristics and: *(i)* complementary metaheuristics; *(ii)* exact methods; *(iii)* constraint programming; or *(iv)* machine learning. The author proposes a general two-level classification. In this sense, it is possible to distinguish between low-level hybridizations, in which a given internal function of a metaheuristic is replaced by another optimization method, and high-level hybridizations, where the different optimization methods are self-contained. In a second phase, these algorithms can be further classified into relay or teamwork hybridization. While in the former the techniques are applied one after another (each using the output of the previous as its input), the latter represents cooperative optimization models. In Jourdan et al. (2006), authors describe applications of data mining techniques to help metaheuristics. Finally, a survey on the integration of machine learning in evolutionary computation can be found in Zhang et al. (2011). The work presented in Corne et al. (2012) gathers the synergies between OR and data

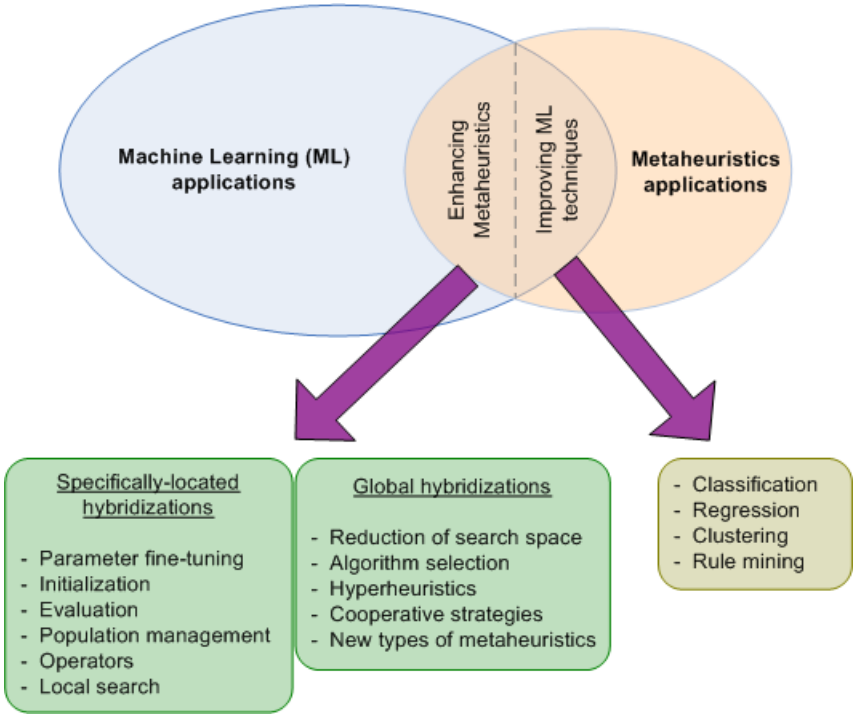


FIGURE A.4: Classification of works combining metaheuristics and machine learning.

mining, remarking the growing importance of multi-objective approaches. The authors highlight three benefits of employing data mining in OR: *(i)* increasing the quality of the results of OR algorithms; *(ii)* speeding up OR algorithms; and *(iii)* selecting an OR algorithm based on instance properties.

Our work builds on the classification in Jourdan et al. (2006) and extends it by proposing more categories and analyzing a higher number of works. In our view, the classification in Jourdan et al. (2006) is more suitable for works where machine learning is employed to enhance metaheuristics than the one presented in Talbi (2013), which was designed to be more general and to include other hybridizations. In particular, works are classified into specifically-located hybridizations (where machine learning is applied in a specific procedure) and global hybridizations (in which machine learning has a higher effect on the metaheuristic design). As part of the first group, the following categories are defined: parameter fine-tuning, initialization, evaluation, population management, operators, and local search. On the other hand, the second one includes: reduction of search space, algorithm selection, hyperheuristics, cooperative strategies, and new types of metaheuristics.

Similarly, there are a few reviews on works where metaheuristics are used to improve the performance of machine learning techniques. For instance, Freitas (2008) focuses on two evolutionary algorithms (EAs), namely GAs and genetic programming (GP), and discusses their application to discovery of classification rules, clustering, attribute selection and attribute construction. Corne et al. (2012) analyzes the role of OR in data mining discussing the relevance of exact methods, heuristics and metaheuristics in supervised classification, unsupervised classification, rule mining and feature selection. More recently, Dhaenens and Jourdan (2016) provides an overview of the use of optimization in Big Data, focusing on metaheuristics. The book introduces the role of metaheuristics in clustering, association rules, classification, and feature selection in classification. It also includes a chapter listing all available frameworks for metaheuristics, data mining, and the combination of both. Building on these reviews, we arrange the literature works into the following categories: classification, regression, clustering, and rule mining.

Figure A.4 shows the classification scheme we use. Some relevant and representative works, both considering machine learning for enhancing metaheuristics and metaheuristics in machine learning, are reviewed in Sections 3 and 4, respectively.

3. Using machine learning for enhancing metaheuristics

In order to improve clarity, the review on how machine learning techniques have been used to enhance metaheuristics has been divided into two sub-sections: the first one analyzes local-level hybridizations while the second one discusses global-level hybridizations. Each of these, in turn, have been classified by the corresponding topic.

Specifically-located hybridizations

The ***fine-tuning of metaheuristic parameters*** is known to have a significant effect on the algorithm performance. However, this issue is not always properly addressed and many researchers still continue selecting parameter values by performing exhaustive testing or copying values recommended for similar instances or problems.

Basically, there are three approaches:

1. *Parameter control strategies* (De Jong, 2007) apply a dynamic fine-tuning of the parameters by controlling and adapting the parameter values during the solving of an instance. The main types of control are: *(i)* deterministic, which modifies the parameter values by some deterministic rule; and *(ii)* adaptive, which employs feedback from the search. For instance, there are works relying on fuzzy logic (Jeong et al., 2009), support vector machine (SVM) (Zennaki and Ech-Cherif, 2010), and linear and SVM regression (Lessmann et al., 2011).

2. *Parameter tuning strategies* assume that the algorithms are robust enough to provide good results for a set of instances of the same problem with a fixed set of parameter values. Frequently, researchers focus on a subset of the instances and analyze their fitness landscapes. Popular techniques are: response surface (Gunawan et al., 2013), logistic regression (Ramos et al., 2005), and tree-based regression (Bartz-Beielstein et al., 2004).

3. *Instance-specific parameter tuning strategies* present characteristics from the previous approaches. While the parameter values are constant as in the second approach, they are specific for each instance as in the first. These strategies employ a learning mechanism able to return recommended sets of parameter values given a number of instance features. Techniques employed are: Bayesian networks (Pavón et al., 2009), case-based reasoning (CBR) (Pereira et al., 2013), fuzzy logic (Ries et al., 2012), linear regression (Caserta and Rico, 2009), and neural networks (Dobslaw, 2010).

A highly popular approach related to the first category is known as reactive search (Battiti and Brunato, 2010). It proposes the integration of sub-symbolic machine learning techniques into heuristics, in order to allow the algorithm for self-tuning.

Typically, metaheuristics generate their ***initial solutions*** randomly, using design of experiments (Leung and Wang, 2001), or via a fast heuristic. There are also works employing machine learning techniques. For instance, some of them apply CBR to initialize GAs (Ramsey and Grefenstette, 1993; Louis and McDonnell, 2004; Li et al., 2011c), while others explore the use of Hopfield neural networks (Yalcinoz and Altun, 2001). In De Lima et al. (2008) the authors suggest using the Q-learning algorithm in the constructive phase of a GRASP and a reactive GRASP metaheuristics. In this line, the hybridization of data mining and the GRASP metaheuristic is discussed in Santos et al. (2008).

In real-life applications it is common to find objective functions and constraints that are computationally expensive to ***evaluate*** (Lim et al., 2010; Tenne and Goh, 2010). In these cases, it is required to build an approximation model to assess solutions employing polynomial regression (Zhou et al., 2005), neural networks (Hunger and Huttner, 1999; Adra et al., 2005; Pathak et al., 2008), SVM (Yang et al., 2009), Markov fitness models (Brownlee et al., 2010), kriging (Díaz-Manríquez et al., 2011) or radial basis functions (Regis, 2014), for example. Some authors combine their use with that of real objective functions (Rasheed and Hirsh, 2000; Zhou and Zhang, 2010). An interesting survey of model approximation in evolutionary computation may be found in Jin (2005). Another option to reduce evaluation costs is to evaluate only representative solutions. Following this idea, in Yoo and Cho (2004) the authors apply fuzzy clustering. Similarly, in Jin and Sendhoff (2004) the authors suggest using clustering techniques and neural networks ensembles.

Regarding ***population management***, many authors attempt to extract information from solutions already visited and employ it to build new ones, aiming to explore more promising search

spaces. A number of works rely on the Apriori algorithm (to identify interesting subsolutions) (Dalboni et al., 2003; Santos et al., 2005; Ribeiro et al., 2006; Santos et al., 2006) or on CBR (Louis, 2003). Another important issue in PMs is the population diversity, since maintaining it may lead to better performances. The most common technique for promoting diversity is clustering analysis. In Streichert et al. (2003), for instance, individuals in a GA are separated in different sub-populations based on their features and only those in the same cluster compete for survival. The selection operator is applied independently to each cluster. In contrast, in Aichholzer et al. (2002) the authors allow interactions among sub-populations of an evolutionary strategy when selecting candidates for recombination. Other works relying on clustering analysis, but in the context of multi-objective metaheuristics, are Pulido and Coello (2004) and Park and Lee (2009).

The search of a metaheuristic may be improved by introducing knowledge in ***operators*** such as mutation or crossover operators in PMs. For example, Handa et al. (2002) propose a coevolutionary GA that incorporates an extraction mechanism to be employed in the crossover. Two classification algorithms are tested: C4.5 and CN2. In Michalski (2000), the authors design a class of evolutionary computation processes called learnable evolution model (LEM), which uses symbolic learning methods to create rules that explain why certain individuals are superior to others. These rules are then employed to create new populations by avoiding past failures, using recommendations or generating variants. In Jourdan et al. (2005), this class is extended to address multi-objective problems seeking rules to identify why some individuals dominate others.

Some machine learning techniques have been used as ***local searches***. For instance, Gaspar-Cunha and Vieira (2004) employ a multi-objective EA combined with an inverse neural network. This neural network is a local search aiming to discover better individuals from previous generations. In particular, it is trained considering parameters and criteria as inputs and outputs, respectively. First, the criteria obtained from individuals of the present generation are slightly modified. Then, the parameters for the new individuals are obtained using the neural network in a reverse way. The authors test their approach on a set of benchmark bi-objective functions. A similar approach is suggested in Adra et al. (2005) to be applied to an aircraft control system design application.

Global hybridizations

A few works have attempted to ***reduce the search space*** in order to make more effective and efficient searches. Machine learning techniques used are: clustering techniques (Hu and Huang, 2004; Senjyu et al., 2005; Barreto et al., 2007; Adibi and Shahrabi, 2013), neural networks (ChangYoon and Way, 2001; Marim et al., 2003) and principal component analysis (Auger and Hansen, 2005).

The ***algorithm selection problem*** (ASP) aims to predict the algorithm from a portfolio that will perform best, employing a given set of instance features. The framework for this problem was initially proposed by Rice (1976), where it was applied to partial differential equation solvers. More recently, Smith-Miles (2009) presents it in the context of optimization algorithms. There are four basic elements in the framework: *(i)* the problem space P represents the set of problem instances; *(ii)* the feature space F includes instance characteristics; *(iii)* the algorithm space A is the portfolio of available algorithms; and *(iv)* the performance space Y is the mapping of each algorithm to the performance metrics. Accordingly, the ASP can be stated as follows (Smith-Miles et al., 2014): given a problem instance $x \in P$ with feature vector $f(x) \in F$, the ASP searches the selection mapping $S(f(x))$ into algorithm space A such that the selected algorithm $\alpha \in A$ maximizes the performance mapping $y(\alpha, x) \in Y$. Thus, for instance, Smith (2008) develops a methodology to predict the performance of metaheuristics and acquire insights into the relation between search space characteristics of an instance and algorithm performance. The author tests the ILS and the robust TS metaheuristics, as well as the max-min ant system for solving the quadratic assignment problem. A neural network implementing genetic adaptive smoothing parameter selection is trained to predict which algorithm will perform best. Also, in Kanda et al. (2011), an approach is designed to select the best optimization method for solving a given travelling salesman problem (TSP) instance. Initially, 14 TSP properties and the performance values obtained with each metaheuristic analyzed (GRASP, TS, SA and GA) are stored. Then, a rank of metaheuristics is determined by using a multi-layer perceptron network. Several network architectures are assessed. In Smith-Miles et al. (2014), the authors construct a methodology to compare the strengths and weaknesses of a set of optimization algorithms. First, the instance space is generated. This step includes selecting a subset of features to obtain a two-dimensional instance space (for a better

visualization) and provide a good separation of easy and hard instances. Afterwards, classification techniques are used to identify the regions where an algorithm performs well or poorly. Finally, an analysis of the algorithmic power is performed considering the size and location of each algorithm footprint. The experiment is carried out with 8 algorithms for solving the graph coloring problem.

According to Burke et al. (2010), ***hyperheuristics*** may be described as search methods or learning mechanisms for selecting or generating heuristics to solve computational search problems. Typically, these methods do not aim to obtain better results than problem-specific metaheuristics, but to be able to automate the design of heuristic methods and/or deal with a wide range of problems. The authors propose a basic classification, which takes into account the following dimensions: *(i)* the nature of the heuristic search space (either heuristic selection or generation); and *(ii)* the feedback, since hyperheuristics may learn (following online or offline learning strategies) or not. Whereas online learning refers to methods that learn during the solving of a problem instance, offline learning methods try to extract information from a set of training instances to be applied for solving new instances. A comprehensive survey on hyper-heuristics may be found in Burke et al. (2013). In Thabtah and Cowling (2008), the authors explore the potential of associative classifiers in a hyperheuristic approach for solving the training scheduling problem. The classifiers have to choose the low-level heuristic (which represents a given local search neighborhood) to employ at each step while constructing a solution. Reinforcement learning is highly popular in methodologies selecting heuristics employing an online learning strategy (e.g., see Berberoğlu and Uyar, 2010). In this case, each heuristic has associated a score that determines the probability of being selected. These scores are updated according to the intermediate solutions obtained with each heuristic. There are also a number of works employing regression techniques. For instance, related to the evaluation category, Li et al. (2011a) suggest employing neural networks and logistic regression to predict objective function values of solutions in a hyperheuristic search. It is also worth mentioning the approach described in Burke et al. (2006), where CBR is used for selecting heuristics when addressing course and exam timetabling problems. In Ortiz-Bayliss et al. (2013), the authors address a constraint satisfaction problem, where the order in which the variables are selected affects the complexity of the search. The authors present a hyperheuristic based on a logistic regression model that decides which variable ordering heuristic should be applied given the features of an instance at different steps of the search. In Asta and Ozcan (2014) an apprenticeship learning hyperheuristic is proposed for vehicle routing. Taking a state-of-the-art hyperheuristic as an expert, the authors follow a learning approach that yields various classifiers, which capture different actions that the expert performs during the search. While this approach relies on a C4.5 algorithm, in Tyasnurita et al. (2015) it is improved by using a multilayer perceptron. Another approach is presented in Asta et al. (2016), where a tensor-based online learning selection hyperheuristic is designed for nurse rostering. The proposed approach consists of the consecutive iteration of four stages: during the first and second stage, two tensors are constructed considering different heuristic selection and move acceptance methods; at the end of the second stage, each tensor is subjected to factorization and, using the information of both tensors, the heuristic space is partitioned; the third is a parameter control phase for the heuristics; and the final stage performs the search switching between heuristics periodically, using appropriate heuristic parameter values.

During the last decades, a new trend in optimization has emerged as a consequence of the technological development based on ***cooperative strategies***. It consists in combining several algorithms/agents to produce a hybrid strategy in which they cooperate in parallel or sequentially. Communication among them can be either many-to-many (direct) or memory-based (indirect). Agents may share partial or complete solutions and models, among others. It is broadly accepted that strategies based on agents with unrestricted access to shared information may experiment premature convergence. Commonly, there is an agent that coordinates the search of the others, organizing the communication. This strategy attempts to develop a robust methodology that provides high-quality solutions by exploiting the specific advantages of each algorithm. For example, Cadenas et al. (2009) develop a centralized hybrid metaheuristic cooperative strategy, where knowledge is incorporated into the coordinator agent through fuzzy rules. These rules have been defined from a knowledge extraction process applied to the results obtained by each metaheuristic. Then, the coordinator agent collects information and sends orders to each solver agent that will affect its search behaviour (such as re-initiate the search with a specific initial solution, or change the parameter values). The strategy is tested on the knapsack problem, employing a TS, a SA, and a GA. In Asta (2015), the author describes an approach to deal with the permutation flow shop

scheduling problem (PFSP), where a set of agents run in parallel, each applying a given metaheuristic. The best solutions are stored and employed to form a tensor. This tensor is factorized and the pattern obtained is sent to all agents, which will use it to try to build better solutions. In Martin et al. (2016), a cooperative strategy relying on different metaheuristic / local search combinations is put forward. The architecture makes use of two types of agents: the launcher and the metaheuristic agent. The launcher conducts the following tasks: queue instances, configure other agents and gather solutions. Metaheuristic agents execute one of the metaheuristic / local search heuristic combinations. Each of them continuously adapts itself according to a cooperation protocol based on reinforcement learning and pattern matching. This proposal is tested on the PFSP and the capacitated vehicle routing problem.

There are several ***new metaheuristics*** based on learning procedures. Most rely on the fact that a set of pseudo-optimal solutions may be considered a sample drawn from an unknown probability distribution. This distribution may be estimated by employing a selected set of promising solutions and used to generate new solutions. A review of these metaheuristics, called estimation of distribution algorithms (EDAs), can be found in Pelikan et al. (2002). Typically, authors employ these algorithms using fixed-length strings over a finite alphabet to represent solutions. They may be classified into three groups depending on whether no interactions are considered, or only pairwise or multiple ones. From the first group, the most popular are: the population-based incremental learning (PBIL) (Baluja, 1994), the compact genetic algorithm (cGA) (Harik, 1999), and the univariate marginal distribution algorithm (UMDA) (Mühlenbein and Paass, 1996). Some well-known algorithms assuming only pairwise interactions are: the mutual-information-maximizing input clustering (MIMIC) (De Bonet et al., 1997) algorithm and the bivariate marginal distribution algorithm (BMDA) (Pelikan and Mühlenbein, 1999). Considering multiple interactions, there are: the extended compact genetic algorithm (ECGA) (Harik et al., 1999), the factorized distribution algorithm (FDA) (Mühlenbein et al., 1999), and the Bayesian optimization algorithm (BOA) (Pelikan et al., 2000). These metaheuristics have been employed in a wide range of fields such as routing (Euchi, 2014; Wang et al., 2015), scheduling (Ceberio et al., 2012), and nutrition (Gumustekin et al., 2014).

4. Using metaheuristics to improve machine learning

Metaheuristics have been extensively employed to improve machine learning tasks. Briefly, we review some of the most successful approaches in the supervised learning topic, both in classification and regression, and in the unsupervised learning topic, including clustering and rule mining.

Classification is a popular problem in supervised learning, consisting in identifying to which of a set of categories a new observation belongs, on the basis of a training set of data containing observations whose category membership is known. In this context, metaheuristics have been mainly applied for feature selection, feature extraction and parameter fine-tuning. In Escalante et al. (2016) authors suggest that the bags of visual words algorithm could be improved when non linear combinations of weighted features obtained with GP are considered. The approach has successfully been applied to the object recognition field, learning both the weights of each visual word (feature) and the non linear combination of them. Similar approaches have been presented for large feature sets. Thus, Stein et al. (2005) employ GAs to select discriminant features applied to intrusion detection using a decision trees classifier. Also studying classification trees, Sörensen and Janssens (2003) implements a GA to generate a set of diverse trees, each with a large explanatory power. In Fernández-Caballero et al. (2010), the authors present a multi-classification algorithm relying on multi-layer perceptron neural network models. In order to obtain high levels of sensitivity and accuracy (which may be conflicting measures), a Pareto-based multi-objective optimization methodology based on a memetic EA is proposed. In Huang and Wang (2006), the use of GAs is proposed to simultaneously optimize the parameters of the SVM algorithm and perform a feature selection process. In Garrett et al. (2003), a GA performs feature selection on electroencephalogram signals (EEG) applying non linear classifiers (SVM and neural networks). Working on cancer diagnosis, García-Nieto et al. (2009) selects gene by applying a GA combined with SVM. The approach focuses on sensitivity, specificity and number of genes. A comparison between approaches with different criteria and one relying on the k-means algorithm is put forward. In Yusta (2009) different metaheuristics are evaluated also in the context of feature selection. The authors implement a TS, a memetic algorithm, a GRASP and a GA, and compare their effectiveness with sequential forward feature selection. Different niching GAs

for feature selection are proposed in Aguilera et al. (2007), which are applied to the design of fuzzy rule-based classification systems. In Xue et al. (2012) two multi-objective PSO algorithms are designed for feature selection, which aim to maximize the classification performance while minimizing the number of features. They are compared against several conventional methods and three well-known multi-objective EAs using the k-nearest neighbor algorithm as classifier. In Candelieri (2011), the author employs a GA, a TS and an ACO for parameter fine-tuning of a single classifier and classifiers ensemble optimization, working with SVM.

Regression aims to estimate the relationships among a response variable and one or more explanatory variables, and has a wide range of applications. Typically, the use of machine learning is related to the training of complex regression models. Neuroevolution is an emergent field which employs EAs to train neural networks. Thus, Yao (1999) provides a literature review focusing on elements evolved: connection weights, architectures, learning rules, and input features. In Stanley and Miikkulainen (2002), the authors develop the neuroevolution of augmenting topologies (NEAT) method, which evolves topologies and weights at the same time. They claim that its efficiency resides in: *(i)* employing a principled method of crossover of different topologies; *(ii)* protecting structural innovation using speciation; and *(iii)* incrementally growing from minimal structure. More recently, Turner and Miller (2014) has shown the benefits of optimizing each neuron's transfer function, creating heterogeneous networks. In a similar approach, Carvalho et al. (2011) present a methodology to find the best architecture of a neural network using metaheuristics. The authors tested the following ones: generalized extremal optimization, VNS, SA, and canonical GA.

Clustering refers to grouping a set of objects in such a way that objects in the same group are more similar to each other than to those in other groups. This vague definition encompasses a high number of models. Centroid models are based on an NP-hard optimization problem (thus, only approximated solving methods such as metaheuristics may be employed). In Das et al. (2009) differential evolution algorithms are applied to clustering problems (single and multi-objective). In Selim and Alsultan (1991) a TS metaheuristic is implemented to computationally deal with the non convex optimization problem of the unsupervised clustering of data samples. Similarly, Shelokar et al. (2004) use ACO to cluster objects, obtaining faster results in terms of the number of needed operations (i.e., objective functions evaluations). Other authors use GAs for the same task (De Jong et al., 1993; Chiou and Lan, 2001; Garai and Chaudhuri, 2004). In Marinakis et al. (2008), a PSO metaheuristic is applied to the clustering problem, improving the results obtained using a TS and classic unsupervised learning methods. In Govindarajan et al. (2013), the authors also use a PSO metaheuristic to automatically cluster data from students in a learning management system (LMS) to adapt teaching resources to specific students' needs. Gene clustering is performed in Banu and Andrews (2015), where a comparative study is presented based on the following metaheuristics: GA, PSO, cuckoo search and levy flight cuckoo search. More recently, Ferone et al. (2016) present a GRASP metaheuristic for biclustering (i.e., considering both genes and conditions) of gene expression data. A validation is completed with different synthetic datasets. The reader can find more details in the applications of metaheuristics to unsupervised learning in the surveys (Hruschka et al., 2009; Kurada et al., 2013).

Rule mining gathers methods for discovering relevant relations between variables in large databases. For example, a hybrid approach is presented in Carvalho and Freitas (2002) for discovering small-disjunct rules combining a decision tree algorithm (C4.5) and a GA. While the first is employed for large-disjuncts rules, the metaheuristic works on small ones. This hybrid approach achieves better predictive accuracy. In the book of Freitas (2002), in addition to present data mining tasks and paradigms, the author describes the application of GAs and GP for rule discovery, and EAs for generating fuzzy rules. After modeling association rules discovery as an optimization problem, Khabzaoui et al. (2004) explore the use of a GA to obtain associations between genes from DNA microarray data. Noticing that most approaches tend to seek only frequent rules, Khabzaoui et al. (2008) propose a multi-objective approach combining a GA and exact methods to discover interesting rules in large search spaces. A public micro-array database is employed to carry out computational experiments. A multi-objective metaheuristic approach is proposed in Ishida et al. (2009) to create rules and build a Pareto front considering the sensitivity and specificity criteria. A GRASP with path-relinking is implemented.

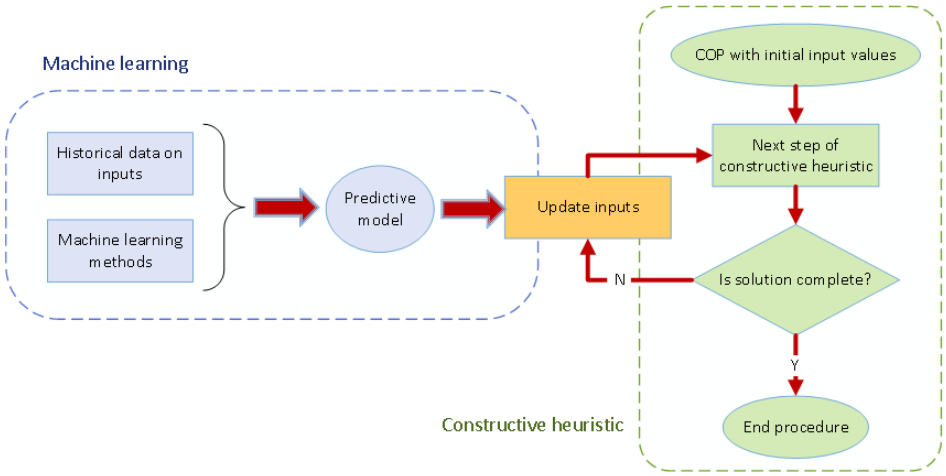


FIGURE A.5: Basic scheme of a learnheuristic framework.

5. Our learnheuristic framework for solving COPDIs

After reviewing in Sections 3 and 4 works where machine learning may enhance metaheuristics or metaheuristics may improve machine learning, this section presents our learnheuristic framework. It integrates both fields, making it possible to address a specific kind of COP.

As previously described, our learnheuristic framework aims at solving COPs in which the model inputs (either located in the objective function or in the set of constraints) are not fixed in advance. Instead, these inputs (e.g., customers' demands, serving times, etc.) might vary in a predictable way according to the current status of the partially-built solution at each iteration of the constructive heuristic. More formally, these problems might be represented as follows:

$$\begin{array}{ll} \text{Min} & C(s, I_{OF}(s)) \quad \text{or, alternatively,} \\ \text{Max} & B(s, I_{OF}(s)) \end{array} \tag{A.1}$$

$$\text{subject to:} \quad Q_j(s, I_C(s)) \leq r_j \quad \forall j \in J \tag{A.2}$$

$$s \in S \tag{A.3}$$

where: *(i)* S refers to a discrete space of possible solutions s; *(ii)* $C(s)$ represents a cost function (alternatively, $B(s)$ represents a benefits function); *(iii)* $I_{OF}(s)$ and $I_C(s)$ refer to inputs in the objective function or the constraints, respectively; and *(iv)* Equations 2 represent a set of constraints. Thus, the aim of this type of problems is to minimize a function of costs (or, alternatively, maximize a function of benefits) subject to a number of constraints. The novel characteristic is that inputs in the objective function and/or the constraints may depend on the solution structure, which makes them to be dynamic as the partially-built solution evolves, and not fixed in advance. This is a basic case of dynamic inputs, which could be easily extended to deal with multi-objective and/or stochastic problems (Yang et al., 2013).

In order to deal with these COPDIs, we propose the use of a learnheuristic framework as explained next in detail. Figure A.5 shows the basic scheme of this approach. Initially, historical data on different system states (e.g., different assignments of users to RATs) and their associated inputs (e.g., users' demands observed for the corresponding assignments) are employed to generate machine-learning predictive models (e.g., regression models, neural network models, etc.). Then, these predictive models are iteratively used during the heuristic-based constructive process in order to obtain updated estimates of the problem inputs (e.g., users' demands) as the structure of the solution (e.g., users-to-RAT assignment map) varies. Eventually, once the construction process is finished, a complete solution is generated. Without the use of the learning mechanism, the heuristic-based construction process will not take into account the variations in the inputs due to changes in the solution structure, which will lead to sub-optimal solutions.

Pseudo-code 7 contains a more detailed description of the basic learnheuristic framework. Notice that the main loop iterates over a list of elements that are provided by the constructive heuristic (e.g., next user-to-RAT assignment). At each iteration, the algorithm evaluates the current

status of the partially-built solution, makes use of the predictive model to update the problem inputs according to this status, and follows the heuristic logic to take another solution-building step based on the new problem inputs. This basic scheme could be extended in different ways, e.g.: *(i)* by employing an online approach, where new inputs are used to update and improve the predictive model; and *(ii)* by using an "assembled" or blending approach, in which several models are built and then combined in order to generate the inputs estimates.

Algorithm 1 Basic scheme of learnheuristic algorithms.

```
Learnheuristics(historicalData, inputs)
% historicalData: historical data on different system states and their associated inputs
% inputs: problem instance
  model ←buildPredictiveModel(historicalData)
  sol ← empty
  while (sol is not completely built) do % iterative learning-heuristic process
      inputs ←updateInputs(model, inputs, sol)
      sol ← nextHeuristicStep(inputs, sol)
  end while
  return sol
```

As any other heuristic procedure, the aforementioned learnheuristic approach can be integrated into a more complex metaheuristic framework. For instance, it can be easily integrated into multi-start, GRASP, or ILS frameworks. In order to do so, the learnheuristic algorithm may be combined with biased-randomization strategies as the ones proposed in Juan et al. (2011b), which allow for generating a number of high quality solutions from a deterministic heuristic (i.e., one that does not include any random behavior, thus providing always the same solution, as opposed to a randomized heuristic). Pseudo-code 15 gives an example of how this integration could be made in the case of a simple multi-start framework.

Algorithm 2 Multi-start metaheuristic integrating a learnheuristic algorithm with biased-randomization.

```
Multi-start(historicalData, inputs, distribution, maxTime)
% distribution: probability distribution and parameters for the biased-randomization process
% maxTime: maximum computing time allowed
  initInputs ← inputs % copy of initial inputs
  elapsedTime ← 0
  initTime ← currentTime
  bestSol ← biasedRandLearnheuristic(historicalData, inputs, distribution)
  inputs ← initInputs % reset inputs
  while (elapsedTime ≤ maxTime) do
      newSol ← biasedRandLearnheuristic(historicalData, inputs, distribution)
      newSol ← localSearch(newSol)
      if (cost(newSol) ≤ cost(bestSol)) then
          bestSol ← newSol
      end if
      inputs ← initInputs % reset inputs
      elapsedTime ← currentTime−initTime
  end while
  return bestSol
```

5. Potential applications in different fields

This section provides a series of examples, belonging to different optimization areas, in which the use of learnheuristics might facilitate the solving process of more realistic and rich models.

- ***Transportation:*** In the transportation area and, in particular, in vehicle and arc routing problems, inputs such as the customers' demands might be dynamic in the sense that they might depend upon the delivery time and whether or not certain time-windows are satisfied.

It is a function of the solution structure, e.g., the order in which the customers are visited, the number and type of vehicles employed, etc. Similarly, the traveling times, which affect the distribution cost, might also be dynamic and dependent on the solution structure, specially in large cities where traffic jams occur frequently.

- ***Logistics:*** As discussed in Calvet et al. (2016d), the assignment of customers to certain distribution centers might have a significant effect on the customers' willingness to spend (i.e., on their demands). Therefore, in realistic facility location problems and similar ones, modelers might have to face dynamic inputs influenced by the shape of the solution (i.e., which facilities are open and how customers are assigned to them).

- ***Production:*** In scheduling problems, for instance, processing times of jobs into machines might not be fixed but, instead, they may be a function of the order in which they are processed by the machine (e.g., due to 'fatigue' issues or to breaks). A similar situation can happen in project scheduling, where some working teams might be more efficient than others and assigning them to a given sub-project could cause the delay of others.

- ***Finance:*** In problems such as portfolio optimization, the covariance matrix that measures the risk associated with each pair of assets could also be a function of the current portfolio structure (i.e., which other assets are already included and which percentage of investment has been assigned to each of them). Likewise, the expected return for each asset might depend on the current composition of the portfolio. This dynamic behavior of the inputs can be extended to different risk-management problems which include some sort of portfolio optimization.

6. A numerical experiment

This section describes a simple numerical experiment that illustrates the use of a learnheuristic approach. We consider a vehicle routing problem in which each customer's demand will depend on the order in which the customer is visited. For each customer, its initial demand value is an upper-bound of the real demand. In other words, this value will be valid only if the customer is visited by a vehicle as the first stop in its route. Then, as the position in which the customer is visited increases, the customer's demand will be reduced (i.e., higher service times imply lower demands). Therefore, if we use a constructive heuristic to solve the vehicle routing problem considering the initial demands as fixed inputs, the solution will be overestimating the real demands. This, in turn, will lead to higher costs, since the number of routes employed to satisfy the real demands will be higher than necessary. Likewise, vehicles will be carrying more load than strictly required. On the contrary, if we are able to forecast the real customers' demands based on their position inside a route, then each route might be able to cover additional customers and the total distance-based costs will be reduced.

In order to compare both cases, the Clarke and Wright constructive heuristic (Clarke and Wright, 1964) have been applied to a random instance belonging to the well known benchmarks for the vehicle routing problem, particularly to the instance P-n70-k10 (http://neo.lcc.uma.es/vrp/wp-content/data/instances/Augerat/P-VRP.zip). This instance consists of 12 vehicles with the same capacity – 135 units – and 70 customers. The customers are scattered in an area with x axis ranging from -34 to 30, and y axis ranging from -36 to 36. The depot is located at coordinates (0,0). Initial customers demands range from 5 to 37.

On the one hand, we have considered fixed demands, i.e., the original demands provided by the instance are used to obtain the solution through the heuristic in the standard way. On the other hand, we have created a predictive model to calculate dynamic demands in order to apply a learnheuristic algorithm following the scheme in Pseudo-code 7. In this case, for illustrative purposes, the following linear regression model has been considered:

$$d = \max\{k_1 \cdot d_0,\ d_0 - k_2 \cdot d_0 \cdot (p-1)\} \tag{A.4}$$

where d is the predicted demand of a given customer, d_0 is the initial demand of the same customer, k_1 and $k_2 \in (0, 1)$, and p is the position order in the route of the aforementioned customer. In particular, we have applied $k_1 = 0.20$ and $k_2 = 0.05$. The regression model aims at predicting a customer's demand taking into account the position in which the customer is served in the route, so that the demand decreases as the position increases or until a certain demand lower-bound is

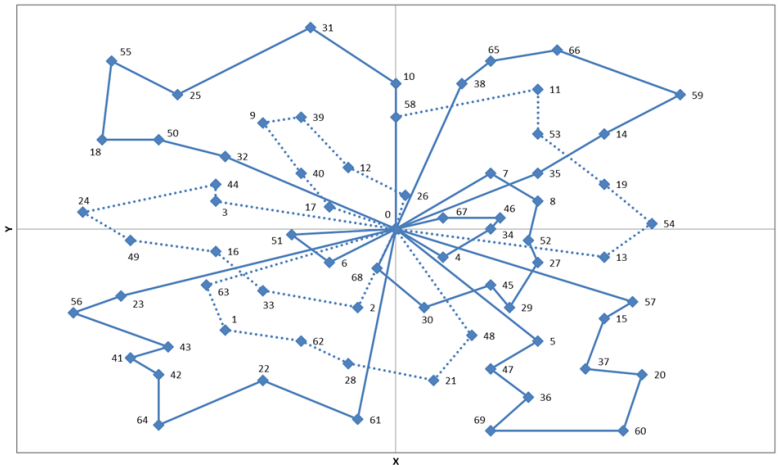


FIGURE A.6: Routes obtained considering fixed demands.

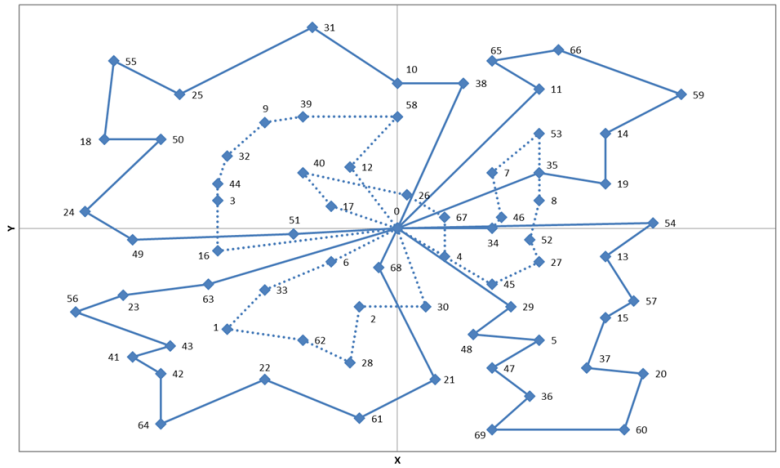


FIGURE A.7: Routes obtained considering dynamic demands.

reached. Once we have the model, the heuristic starts the loop building routes until a solution is obtained. Thus, each time the heuristic performs a step, incorporating a new customer in a route or moving a customer from one route to another, the customer's demand is predicted and updated according to its new position in the corresponding route. As mentioned before, the total demand in a route is limited by the capacity of the vehicle. Therefore, this prediction affects the next steps that can be performed.

Accordingly, we have performed two experiments using these two versions of the Clarke and Wright heuristic, both in its standard way and using the learnheuristic framework. When fixed demands are considered, the best solution the constructive heuristic is able to obtain has an associated cost of 896.86, and it involves 11 routes (see Figure A.6). However, if demands are predicted taking into account the delivery order, the same heuristic obtains a solution with 8 routes and a cost of 791.26 (see Figure A.7). Therefore, the savings might be noticeable when dynamic demands are considered.

7. Conclusions and future research

Real-life problems faced by companies are becoming increasingly complex. This can be due, among other factors, to the existence of more competitive markets as well as to larger and more interconnected supply chain systems. On the other hand, the technological development of the last few decades allows the implementation of more powerful and faster algorithms, and the analysis of huge amounts of diverse types of data. As a consequence, hybrid approaches for addressing hard combinatorial optimization problems (COPs) are highly popular.

This paper has focused on the combination of metaheuristics and machine learning. An overview of the different approaches and a general classification have been provided. We have presented a specific type of realistic COP which requires this hybridization. In particular, these problems are characterized by inputs (located either in the objective function or the set of constraints) that are not fixed in advance, but may vary according to the solution characteristics. Then we propose a new solving approach, learnheuristic algorithms, to cope with these dynamic optimization problems. This approach relies on machine learning techniques to learn the relationships between inputs and solution characteristics from historical data, and a constructive heuristic (which may be embedded in a metaheuristic algorithm), to build a high quality solution using predictions. Different extensions of this approach can be considered, e.g.: *(i) online version*, in which information regarding new inputs can be used to improve the predictive model and *(ii) blended version*, in which predictions from several models are averaged, not necessarily giving the same weight to each of them. Finally, some potential applications to a variety of fields have been also pointed out.

From the work developed, we foresee several open research lines that could be explored: *(i)* implement the methodologies proposed to specific fields/problems testing several techniques of machine learning; *(ii)* extend the methodology to stochastic and/or multi-objective optimization problems; and *(iii)* study the use of distributed and parallel computing paradigms to allow for real-time decision-making.

Acknowledgments

This work has been partially supported by the Spanish Ministry of Economy and Competitiveness and FEDER (TRA2013-48180-C3-P, DPI2013-44461-P, TIN2015-66951-C2-2-R), and the Catalan Government (2014-CTP-00001). Likewise, we want to thank the support of the UOC doctoral programme.

References

Adibi, M. A. and J. Shahrabi (2013). "A clustering-based modified variable neighborhood search algorithm for a dynamic job shop scheduling problem". In: *The International Journal of Advanced Manufacturing Technology* 70.9, pp. 1955–1961.

Adra, S. F., A. I. Hamody, I. Griffin, and P. J. Fleming (2005). "A hybrid multi-objective evolutionary algorithm using an inverse neural network for aircraft control system design." In: *Congress on Evolutionary Computation*. IEEE, pp. 1–8.

Aguilera, J. J., M. Chica, M. J. del Jesus, and F. Herrera (2007). "Niching genetic feature selection algorithms applied to the design of fuzzy rule-based classification systems". In: *2007 IEEE International Fuzzy Systems Conference*. IEEE, pp. 1–6.

Aichholzer, O., F. Aurenhammer, B. Brandstatter, T. Ebner, H. Krasser, Ch. Magele, M. Muhlmann, and W. Renhart (2002). "Evolution strategy and hierarchical clustering". In: *IEEE T. Magn.* 38.2, pp. 1041–1044.

Asta, S. (2015). "Machine learning for improving heuristic optimisation". PhD thesis. The University of Nottingham, UK.

Asta, S. and E. Ozcan (2014). "An apprenticeship learning hyper-heuristic for vehicle routing in HyFlex". In: *2014 IEEE Symposium on Evolving and Autonomous Learning Systems (EALS)*, pp. 65–72.

Asta, S., E. Ozcan, and T. Curtois (2016). "A tensor based hyper-heuristic for nurse rostering". In: *Knowl.-based Syst.* 98, pp. 185–199.

Auger, A. and N. Hansen (2005). "Performance evaluation of an advanced local search evolutionary algorithm". In: *2005 IEEE congress on evolutionary computation*. Vol. 2. IEEE, pp. 1777–1784.

Baluja, S. (1994). *Population-based incremental learning. A method for integrating genetic search based function optimization and competitive learning*. Tech. rep. DTIC Document.

Banu, P. K. Nizar and S. Andrews (2015). "Gene clustering using metaheuristic optimization algorithms". In: *International Journal of Applied Metaheuristic Computing* 6.4, pp. 14–38.

Barber, D. (2012). *Bayesian reasoning and machine learning*. Cambridge University Press. ISBN: 0521518148, 9780521518147.

Barreto, S., C. Ferreira, J. Paixao, and B. Sousa (2007). "Using clustering analysis in a capacitated location-routing problem". In: *Eur. J. Oper. Res.* 179.3, pp. 968–977.

Bartz-Beielstein, T., K. E. Parsopoulos, and M. N. Vrahatis (2004). "Design and analysis of optimization algorithms using computational statistics". In: *Applied Numerical Analysis & Computational Mathematics* 1.2, pp. 413–433.

Battiti, R. and M. Brunato (2010). "Reactive search optimization: learning while optimizing". In: *Handbook of Metaheuristics*. Springer, pp. 543–571.

Berberoğlu, A. and A. Uyar (2010). "A hyper-heuristic approach for the unit commitment problem". In: *European Conference on the Applications of Evolutionary Computation*. Springer, pp. 121–130.

Boussaïd, I., J. Lepagnot, and P. Siarry (2013). "A survey on optimization metaheuristics". In: *Information Sciences* 237, pp. 82–117.

Brownlee, A. E. I., O. Regnier-Coudert, J. A. W. McCall, and S. Massie (2010). "Using a Markov network as a surrogate fitness function in a genetic algorithm". In: *IEEE Congress on Evolutionary Computation*. IEEE, pp. 1–8.

Burke, E. K., S. Petrovic, and R. Qu (2006). "Case-based heuristic selection for timetabling problems". In: *J. Sched.* 9.2, pp. 115–132.

Burke, E. K., M. Hyde, G. Kendall, G. Ochoa, E. Özcan, and J. R. Woodward (2010). "A classification of hyper-heuristic approaches". In: *Handbook of metaheuristics*, pp. 449–468.

Burke, E. K., M. Gendreau, M. Hyde, G. Kendall, G. Ochoa, E. Özcan, and R. Qu (2013). "Hyper-heuristics: a survey of the state of the art". In: *Eur. J. Oper. Res.* 64.12, pp. 1695–1724.

Caceres, J., P. Arias, D. Guimarans, D. Riera, and A. A. Juan (2014). "Rich vehicle routing problem: a survey". In: *ACM Computing Surveys* 47.2. ISSN: 0360-0300.

Cadenas, J. M., M. C. Garrido, and E. Muñoz (2009). "Using machine learning in a cooperative hybrid parallel strategy of metaheuristics". In: *Inform. Sciences* 179.19, pp. 3255–3267.

Calvet, L., V. Fernandez-Viagas, J. Framinan, and A. A. Juan (2016c). "Combining simulation with metaheuristics in distributed scheduling problems with stochastic processing times". In: *Proceedings of the 2016 Winter Simulation Conference*. Washington D. C., USA, pp. 2347–2357.

Calvet, L., D. Wang, and A. A. Juan (submitted[a]). "A simheuristic algorithm for the stochastic multi-depot vehicle routing problem". In: *International Transactions in Operational Research*.

Candelieri, A. (2011). "A hyper-solution framework for classification problems via metaheuristic approaches". In: *4OR* 9.4, pp. 425–428.

Carvalho, A. R., F. M. Ramos, and A. A. Chaves (2011). "Metaheuristics for the feedforward artificial neural network architecture optimization problem". In: *Neural Computing and Applications* 20.8, pp. 1273–1284.

Carvalho, D. R. and A. A. Freitas (2002). "A genetic algorithm for discovering small disjunct rules in data mining". In: *Appl. Soft Comput.* 2.2, pp. 75–88.

Caserta, M. and E. Quiñonez Rico (2009). "A cross entropy-Lagrangean hybrid algorithm for the multi-item capacitated lot-sizing problem with setup times". In: *Computers & OR* 36.2, pp. 530–548.

Ceberio, J., E. Irurozki, A. Mendiburu, and J. A. Lozano (2012). "A review on estimation of distribution algorithms in permutation-based combinatorial optimization problems". In: *Pattern Recogn.* 1.1, pp. 103–117.

ChangYoon, L. and K. Way (2001). "Reliability optimization design using a hybridized genetic algorithm with a neural-network technique". In: *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences* 84.2, pp. 627–637.

Chiou, Y.-C. and L. W. Lan (2001). "Genetic clustering algorithms". In: *Eur. J. Oper. Res.* 135.2, pp. 413–427.

Clarke, G. and J. Wright (1964). "Scheduling of vehicles from a central depot to a number of delivering points". In: *Operations Research* 12, pp. 568–581.

Corne, D., C. Dhaenens, and L. Jourdan (2012). "Synergies between operations research and data mining: The emerging use of multi-objective approaches". In: *Eur. J. Oper. Res.* 221.3, pp. 469–479.

Dalboni, F. L., L. S. Ochi, and L.M.A. Drummond (2003). "On improving evolutionary algorithms by using data mining for the oil collector vehicle routing problem". In: *International Network Optimization Conference*, pp. 182–188.

Das, S., A. Abraham, and A. Konar (2009). "Metaheuristic pattern clustering–an overview". In: *Metaheuristic Clustering*. Springer, pp. 1–62.

De Bonet, J. S., C. L. Isbell, and P. Viola (1997). "MIMIC: Finding optima by estimating probability densities". In: *Advances in neural information processing systems*. Morgan Kaufmann publishers, pp. 424–430.

De Jong, K. (2007). "Parameter setting in EAs: a 30 year perspective". In: *Parameter setting in evolutionary algorithms*. Springer, pp. 1–18.

De Jong, K. A., W. M. Spears, and D. F. Gordon (1993). "Using genetic algorithms for concept learning". In: *Genetic Algorithms for Machine Learning*. Springer, pp. 5–32.

De Lima, Francisco Chagas, Jorge Dantas De Melo, and Adrião Duarte Doria Neto (2008). "Using the Q-learning algorithm in the constructive phase of the GRASP and reactive GRASP metaheuristics". In: pp. 4169–4176. ISBN: 9781424418213.

Dhaenens, C. and L. Jourdan (2016). *Metaheuristics for Big Data*. Wiley. ISBN: 9781119347606.

Díaz-Manríquez, A., G. Toscano-Pulido, and W. Gómez-Flores (2011). "On the selection of surrogate models in evolutionary optimization algorithms". In: *2011 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, pp. 2155–2162.

Dobslaw, F. (2010). "A parameter tuning framework for metaheuristics based on design of experiments and artificial neural networks". In: *World Academy of Science, Engineering and Technology* 64, pp. 213–216.

Dorigo, M. (1992). "Optimization, learning and natural algorithms". PhD thesis. Politecnico di Milano, Italy.

Escalante, H. J., V. Ponce-López, S. Escalera, X. Baró, A. Morales-Reyes, and J. Martínez-Carranza (2016). "Evolving weighting schemes for the bag of visual words". In: *Neural Comput. Appl.* 0, pp. 1–15.

Euchi, J. (2014). "Hybrid estimation of distribution algorithm for a multiple trips fixed fleet vehicle routing problems with time windows". In: *International Journal of Operational Research* 21.4, p. 433. ISSN: 1745-7645.

Farmer, J. D., N. H. Packard, and A. S. Perelson (1986). "The immune system, adaptation, and machine learning". In: *Phys. D* 2.1-3, pp. 187–204.

Feo, T. A. and M. G. C. Resende (1989). "A probabilistic heuristic for a computationally difficult set covering problem". In: *Oper. Res. Lett.* 8.2, pp. 67–71.

Feo, T. A and M. G. C. Resende (1995). "Greedy randomized adaptive search procedures". In: *Journal of global optimization* 6.2, pp. 109–133.

Fernández-Caballero, J. C., F. J. Martinez, C. Hervas, and P. A. Gutierrez (2010). "Sensitivity versus accuracy in multiclass problems using memetic pareto evolutionary neural networks". In: *IEEE T. Neural Network.* 21.5, pp. 750–770.

Ferone, D., A. Facchiano, A. Marabotti, and P. Festa (2016). "A new GRASP metaheuristic for biclustering of gene expression data". In: *PeerJ PrePrints*.

Freitas, A. (2002). "Data mining and knowledge discovery with evolutionary algorithms". In: *Advances in Evolutionary Computation* 105, pp. 819–845.

— (2008). "A review of evolutionary algorithms for data mining". In: *Soft Computing for Knowledge Discovery and Data Mining*. Springer, pp. 79–111.

Garai, G. and B. B. Chaudhuri (2004). "A novel genetic algorithm for automatic clustering". In: *Pattern Recogn. Lett.* 25.2, pp. 173–187.

García-Nieto, J., E. Alba, L. Jourdan, and E. Talbi (2009). "Sensitivity and specificity based multiobjective approach for feature selection: Application to cancer diagnosis". In: *Inform. Process. Lett.* 109.16, pp. 887–896.

Garrett, D., D. A. Peterson, C. W. Anderson, and M. H. Thaut (2003). "Comparison of linear, nonlinear, and feature selection methods for EEG signal classification". In: *IEEE T. Neur. Sys. Reh.* 11.2, pp. 141–144.

Gaspar-Cunha, A. and Armando S. Vieira (2004). "A hybrid multi-objective evolutionary algorithm using an inverse neural network". In: pp. 25–30.

Gendreau, M. and J.-Y. Potvin (2010). *Handbook of metaheuristics*. 2nd. Springer Publishing Company, Incorporated.

Glover, F. (1977). "Heuristics for integer programming using surrogate constraints". In: *Decision Science* 8.1, pp. 156–166.

— (1986). "Future paths for integer programming and links to artificial intelligence". In: *Computers & Operations Research* 13.5, pp. 533–549.

Govindarajan, K., T. S. Somasundaram, and V. S. Kumar (2013). "Particle swarm optimization (PSO)-based clustering for improving the quality of learning using cloud computing". In: *2013 IEEE 13th International Conference on Advanced Learning Technologies*. IEEE, pp. 495–497.

Gumustekin, S., T. Senel, and M. A. Cengiz (2014). "A comparative study on Bayesian optimization algorithm for nutrition problem". In: *J. Food Nutr. Res.* 2.12, pp. 952–958.

Gunawan, A., H. C. Lau, and E. Wong (2013). "Real-world parameter tuning using factorial design with parameter decomposition". In: *Advances in Metaheuristics*. Springer, pp. 37–59.

Handa, H., M. Baba, T. Horiuchi, and O. Katai (2002). "A novel hybrid framework of coevolutionary GA and machine learning". In: *International Journal of Computational Intelligence and Applications* 2.01, pp. 33–52.

Harik, G. (1999). "Linkage Learning via Probabilistic Modeling in the ECGA". In: *Scalable optimization via probabilistic modeling from algorithms to applications* 61.99010, pp. 39–61.

Harik, G. R., F. G. Lobo, and D. E. Goldberg (1999). "The compact genetic algorithm". In: *IEEE T. Evolut. Comput.* 3.4, pp. 287–297.

Holland, J. H. (1962). "The compact genetic algorithm". In: *Journal of the ACM* 3.9, pp. 297–314.

Hruschka, E. R., R. J. G. B. Campello, and A. A. Freitas (2009). "A survey of evolutionary algorithms for clustering". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 39.2, pp. 133–155.

Hu, X.-B. and X.-Y. Huang (2004). "Solving TSP with characteristic of clustering by ant colony algorithm". In: *Acta Simulata Systematica Sinica* 12, p. 014.

Huang, C.-L. and C.-J. Wang (2006). "A GA-based feature selection and parameters optimization for support vector machines". In: *Expert Syst. Appl.* 31, pp. 231–240.

Hunger, J. and G. Huttner (1999). "Optimization and analysis of force field parameters by combination of genetic algorithms and neural networks". In: *J. Comput. Chem.* 20.4, pp. 455–471.

Ishida, C. Y., A. Pozo, E. Goldbarg, and M. Goldbarg (2009). "Multiobjective optimization and rule learning: Subselection algorithm or meta-heuristic algorithm?" In: *Innovative applications in data mining*. Springer, pp. 47–70.

Jeong, S.-J., K.-S. Kim, and Y.-H. Lee (2009). "The efficient search method of simulated annealing using fuzzy logic controller". In: *Expert Systems with Applications* 36.3, pp. 7099–7103.

Jin, Y. (2005). "A comprehensive survey of fitness approximation in evolutionary computation". In: *Soft. Comput.* 9.1, pp. 3–12.

Jin, Y. and B. Sendhoff (2004). "Reducing fitness evaluations using clustering techniques and neural network ensembles". In: *Genetic and Evolutionary Computation Conference*. Springer, pp. 688–699.

Jourdan, L., D. Corne, D. Savic, and G. Walters (2005). "Preliminary investigation of the learnable evolution model for faster/better multiobjective water systems design". In: *International Conference on Evolutionary Multi-Criterion Optimization*. Springer, pp. 841–855.

Jourdan, L., C. Dhaenens, and E.-G. Talbi (2006). "Using datamining techniques to help metaheuristics: a short survey". In: *International Workshop on Hybrid Metaheuristics*. Springer Berlin Heidelberg. Gran Canaria, Spain, pp. 57–69.

Juan, A. A., J. Faulin, J. Jorba, D. Riera, D. Masip, and B. Barrios (2011a). "On the use of Monte Carlo simulation, cache and splitting techniques to improve the Clarke and Wright savings heuristics". In: *Journal of the Operational Research Society* 62, pp. 1085–1097.

Juan, A. A., J. Faulin, S. Grasman, M. Rabe, and G. Figueira (2015a). "A review of simheuristics: extending metaheuristics to deal with stochastic optimization problems". In: *Operations Research Perspectives* 2, pp. 62–72.

Kanda, J. Y., A. C. P. L. F. Carvalho, E. R. Hruschka, and C. Soares (2011). "Using meta-learning to recommend meta-heuristics for the traveling salesman problem". In: *Machine Learning and Applications and Workshops (ICMLA), 2011 10th International Conference on*. Vol. 1. IEEE, pp. 346–351.

Kennedy, J. and R. C. Eberhart (1995). "Particle swarm optimization". In: *Proceedings of the IEEE International Conference on Neural Networks*, pp. 1942–1948.

Khabzaoui, M, C Dhaenens, and E.-G. Talbi (2004). "A multicriteria genetic algorithm to analyze DNA microarray data". In: *Cec2004: Proceedings of the 2004 Congress on Evolutionary Computation, Vols 1 and 2*, pp. 1874–1881.

Khabzaoui, M., C. Dhaenens, and E.-G. Talbi (2008). "Combining evolutionary algorithms and exact approaches for multi-objective knowledge discovery". In: *RAIRO Operations Research* 42.1, pp. 69–83.

Kirkpatrick, S. (1984). "Optimization by simulated annealing: Quantitative studies". In: *Journal of statistical physics* 34.5-6, pp. 975–986.

Kurada, R. R., K. K. Pavan, and A. V. Rao (2013). "A preliminary survey on optimized multiobjective metaheuristic methods for data clustering using evolutionary approaches". In: *International Journal of Computer Science & Information Technology* 5.

Lantz, B. (2013). *Machine learning with R*. Packt Publishing.

Lessmann, S., M. Caserta, and I. M. Arango (2011). "Tuning metaheuristics: a data mining based approach for particle swarm optimization". In: *Expert Systems with Applications* 38.10, pp. 12826–12838.

Leung, Y.-W. and Y. Wang (2001). "An orthogonal genetic algorithm with quantization for global numerical optimization". In: *Trans. Evol. Comp* 5.1, pp. 41–53.

Li, J., E. K. Burke, and R. Qu (2011a). "Integrating neural networks and logistic regression to underpin hyper-heuristic search". In: *Knowl.-based Syst.* 24.2, pp. 322–330.

Li, Z.-Q., H.-L. Zhang, J.-H. Zheng, M.-J. Dong, Y.-F. Xie, and Z.-J. Tian (2011c). "Heuristic evolutionary approach for weighted circles layout". In: *Information and Automation: International Symposium, ISIA 2010, Guangzhou, China, November 10-11, 2010. Revised Selected Papers*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 324–331.

Lim, D., Y. Jin, Y.-S. Ong, and B. Sendhoff (2010). "Generalizing Surrogate-assisted Evolutionary Computation". In: *Trans. Evol. Comp* 14.3, pp. 329–355.

Louis, S. J. and J. McDonnell (2004). "Learning with case-injected genetic algorithms". In: *Trans. Evol. Comp* 8.4, pp. 316–328.

Louis, Sushil J (2003). "Genetic learning from experience". In: *IEEE Congress on Evolutionary Computation*. Vol. 3, pp. 2118–2125.

Maniezzo, V., T. Stützle, and S. Vo (2009). *Matheuristics: Hybridizing metaheuristics and mathematical programming*. 1st. Springer Publishing Company, Incorporated.

Marim, L. R., M. R. Lemes, and A. D. Pino (2003). "Neural-network-assisted genetic algorithm applied to silicon clusters". In: *Phys. Rev. A* 67.3.

Marinakis, Y., M. Marinaki, and N. Matsatsinis (2008). "A stochastic nature inspired metaheuristic for clustering analysis". In: *International Journal of Business Intelligence and Data Mining* 3.1, pp. 30–44.

Martin, O., S. W. Otto, and E. W. Felten (1992). "Large-step Markov chains for the TSP incorporating local search heuristics". In: *Oper. Res. Lett.* 11.4, pp. 219–224.

Martin, S., D. Ouelhadj, P. Beullens, E. Ozcan, A. A. Juan, and E.K. Burke (2016). "A Multi-Agent Based Cooperative Approach To Scheduling and Routing". In: *Eur. J. Oper. Res.* 254.1, pp. 169–178.

Michalski, R. S. (2000). "Learnable evolution model: Evolutionary processes guided by machine learning". In: *Mach. Learn.* 38.1-2, pp. 9–40.

Mladenovic, N. (1995). "A variable neighborhood algorithm: A new metaheuristic for combinatorial optimization". In: *Abstracts of papers presented at Optimization Days*, p. 112.

Mühlenbein, H. and G. Paass (1996). "From recombination of genes to the estimation of distributions I. Binary parameters". In: *International Conference on Parallel Problem Solving from Nature*. Springer, pp. 178–187.

Mühlenbein, H., T. Mahnig, and A. O. Rodriguez (1999). "Schemata, distributions and graphical models in evolutionary optimization". In: *J. Heuristics* 5.2, pp. 215–247.

Ortiz-Bayliss, J. C., H. Terashima-Marín, and S. E. Conant-Pablos (2013). "A supervised learning approach to construct hyper-heuristics for constraint satisfaction". In: *Mexican Conference on Pattern Recognition*. Springer, pp. 284–293.

Park, S.-Y. and J.-J. Lee (2009). "Improvement of a multi-objective differential evolution using clustering algorithm". In: *2009 IEEE International Symposium on Industrial Electronics*. IEEE, pp. 1213–1217.

Pathak, B. K., S. Srivastava, and K. Srivastava (2008). "Neural network embedded multiobjective genetic algorithm to solve non-linear time-cost tradeoff problems of project scheduling". In: *Journal of scientific and industrial research* 67.2, pp. 124–131.

Pavón, R., F. Díaz, R. Laza, and V. Luzón (2009). "Automatic parameter tuning with a Bayesian case-based reasoning system. A case of study". In: *Expert Systems With Applications* 36.2, pp. 3407–3420.

Pelikan, M. and H. Mühlenbein (1999). "The bivariate marginal distribution algorithm". In: *Advances in Soft Computing*. Springer, pp. 521–535.

Pelikan, M., D. E. Goldberg, and E. Cantu-Paz (2000). "Linkage problem, distribution estimation, and Bayesian networks". In: *Evol. Comput.* 8.3, pp. 311–340.

Pelikan, M., D. E. Goldberg, and F. G. Lobo (2002). "A survey of optimization by building and using probabilistic models". In: *Comput. Optim. Appl.* 21.1, pp. 5–20.

Pereira, I., A. Madureira, P. B. M. Oliveira, and A. Abraham (2013). "Tuning meta-heuristics using multi-agent learning in a scheduling system". In: *Transactions on Computational Science XXI*. Springer Berlin Heidelberg, pp. 190–210.

Potvin, J.-Y. and K. A. Smith (2003). "Artificial neural networks for combinatorial optimization". In: *Handbook of Metaheuristics*. Ed. by F. Glover and G. A. Kochenberger. Boston, MA: Springer US, pp. 429–455.

Pulido, G. T. and C. A. C. Coello (2004). "Using clustering techniques to improve the performance of a multi-objective particle swarm optimizer". In: *Genetic and Evolutionary Computation Conference*. Springer, pp. 225–237.

Ramos, I. C. O., M. C. Goldbarg, E. G. Goldbarg, and A. D. D. Neto (2005). "Logistic regression for parameter tuning on an evolutionary algorithm". In: *Evolutionary Computation, 2005. The 2005 IEEE Congress on*. Vol. 2. IEEE, pp. 1061–1068.

Ramsey, C. L. and J. J. Grefenstette (1993). "Case-based initialization of genetic algorithms". In: *Proceedings of the 5th International Conference on Genetic Algorithms*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc., pp. 84–91.

Rasheed, K. and H. Hirsh (2000). "Informed operators: speeding up genetic-algorithm-based design optimization using reduced models". In: *Proceedings of the 2nd Annual Conference on Genetic and Evolutionary Computation*. Morgan Kaufmann Publishers Inc., pp. 628–635.

Regis, R. G. (2014). "Evolutionary programming for high-dimensional constrained expensive black-box optimization using radial basis functions". In: *IEEE Transactions on Evolutionary Computation* 18.3, pp. 326–347.

Ribeiro, M. H., A. Plastino, and S. L. Martins (2006). "Hybridization of GRASP metaheuristic with data mining techniques". In: *Journal of Mathematical Modelling and Algorithms* 5.1, pp. 23–41.

Rice, J. R. (1976). "The algorithm selection problem". In: *Adv. Comput.* 15, pp. 65–118.

Ries, J., P. Beullens, and D. Salt (2012). "Instance-specific multi-objective parameter tuning based on fuzzy logic". In: *European Journal of Operational Research* 218.2, pp. 305–315.

Santos, H. G., L. S. Ochi, E. H. Marinho, and L. M. A. Drummond (2006). "Combining an evolutionary algorithm with data mining to solve a single-vehicle routing problem". In: *Neurocomputing* 70.1, pp. 70–77.

Santos, L. F., M. H. Ribeiro, A. Plastino, and S. L. Martins (2005). "A hybrid GRASP with data mining for the maximum diversity problem". In: *International Workshop on Hybrid Metaheuristics*. Springer, pp. 116–127.

Santos, L. F., S. L. Martins, and A. Plastino (2008). "Applications of the DM-GRASP heuristic: a survey". In: *International Transactions in Operational Research* 15.4, pp. 387–416.

Selim, S. Z. and K. Alsultan (1991). "A simulated annealing algorithm for the clustering problem". In: *Pattern Recogn.* 24.10, pp. 1003–1008.

Senjyu, T., A. Y. Saber, T. Miyagi, K. Shimabukuro, N. Urasaki, and T. Funabashi (2005). "Fast technique for unit commitment by genetic algorithm based on unit clustering". In: *IEE Proceedings-Generation, Transmission and Distribution* 152.5, pp. 705–713.

Shelokar, P. S., V. K. Jayaraman, and B. D. Kulkarni (2004). "An ant colony approach for clustering". In: *Anal. Chim. Acta* 509.2, pp. 187–195.

Smith, J. E. (2008). "Self-adaptation in evolutionary algorithms for combinatorial optimisation". In: *Adaptive and Multilevel Metaheuristics*. Springer, pp. 31–57.

Smith, K. A. (1999). "Neural networks for combinatorial optimization: a review of more than a decade of research". In: *INFORMS J. on Computing* 11.1, pp. 15–34.

Smith-Miles, K., D. Baatar, B. Wreford, and R. Lewis (2014). "Towards objective measures of algorithm performance across instance space". In: *Comput. Oper. Res.* 45, pp. 12–24.

Smith-Miles, K. A. (2009). "Cross-disciplinary perspectives on meta-learning for algorithm selection". In: *ACM Computing Surveys* 41.1, p. 6.

Sörensen, K. (2015). "Metaheuristics—the metaphor exposed". In: *International Transactions in Operational Research* 22.1, pp. 3–18.

Sörensen, K. and G. K. Janssens (2003). "Data mining with genetic algorithms on binary trees". In: *Eur. J. Oper. Res.* 151.2, pp. 253–264.

Stanley, K. O. and R. Miikkulainen (2002). "Evolving neural networks through augmenting topologies". In: *Evol. Comput.* 10.2, pp. 99–127.

Stein, G., B. Chen, A. S. Wu, and K. A. Hua (2005). "Decision tree classifier for network intrusion detection with GA-based feature selection". In: *Proceedings of the 43rd annual Southeast regional conference-Volume 2*. ACM, pp. 136–141.

Streichert, F., H. Ulmer, and A. Zell (2003). "Evolutionary algorithms and the cardinality constrained portfolio selection problem". In: *Operations Research Proceedings 2003, Selected Papers of the International Conference on Operations Research (OR 2003)*.

Talbi, E.-G. (2009). *Metaheuristics: From design to implementation*. New Jersey: John Wiley & Sons.

— (2013). "Combining metaheuristics with mathematical programming, constraint programming and machine learning". In: *4OR* 11.2, pp. 101–150.

Tenne, Y. and C.-K. Goh (2010). *Computational intelligence in expensive optimization problems*. Vol. 2. Springer Science & Business Media.

Thabtah, F. and P. Cowling (2008). "Mining the data from a hyperheuristic approach using associative classification". In: *Expert Syst. Appl.* 34.2, pp. 1093–1101.

Turner, A. J. and J. F. Miller (2014). "NeuroEvolution: evolving heterogeneous artificial neural networks". In: *Evolutionary Intelligence* 7.3, pp. 135–154.

Tyasnurita, R., E. Ozcan, S. Asta, and R. John (2015). "Improving Performance of a Hyperheuristic Using a Multilayer Perceptron for Vehicle Routing". In: *15th Annual Workshop on Computational Intelligence*. Springer. Lancaster, UK.

Wang, J., K. Tang, J. Lozano, and X. Yao (2015). "Estimation of Distribution Algorithm with Stochastic Local Search for Uncertain Capacitated Arc Routing Problems". In: *IEEE T. Evolut. Comput.* 20.c, pp. 1–1.

Xue, B., L. Cervante, L. Shang, and M. Zhang (2012). "A particle swarm optimisation based multi-objective filter approach to feature selection for classification". In: *PRICAI 2012: Trends in Artificial Intelligence: 12th Pacific Rim International Conference on Artificial Intelligence, Kuching, Malaysia, September 3-7, 2012. Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 673–685.

Yalcinoz, T. and H. Altun (2001). "Power economic dispatch using a hybrid genetic algorithm". In: *IEEE Power Engineering Review* 21.3, pp. 59–60.

Yang, S., Q. H. Liu, J. Lu, S. L. Ho, G. Ni, P. Ni, and S. Xiong (2009). "Application of support vector machines to accelerate the solution speed of metaheuristic algorithms". In: *IEEE T. Magn.* 45.3, pp. 1502–1505.

Yang, S., Y. Jiang, and T. T. Nguyen (2013). "Metaheuristics for dynamic combinatorial optimization problems". In: *IMA Journal of Management Mathematics* 24.4, pp. 451–480.

Yao, X. (1999). "Evolving artificial neural networks". In: *Proceedings of the IEEE* 87.9, pp. 1423–1447.

Yoo, S.-H. and S.-B. Cho (2004). "Partially evaluated genetic algorithm based on fuzzy c-means algorithm". In: *International Conference on Parallel Problem Solving from Nature*. Springer, pp. 440–449.

Yusta, S. C. (2009). "Different metaheuristic strategies to solve the feature selection problem". In: *Pattern Recogn. Lett.* 30.5, pp. 525–534.

Zennaki, M. and A. Ech-Cherif (2010). "A new machine learning based approach for tuning metaheuristics for the solution of hard combinatorial Optimization problems". In: *Journal of Applied Sciences* 10, pp. 1991–2000.

Zhang, J., Z.-H. Zhang, Y. Lin, N. Chen, Y.-J. Gong, J.-H. Zhong, H. Chung, Y. Li, and Y.-H. Shi (2011). "Evolutionary Computation Meets Machine Learning: A Survey". In: *Comp. Intell. Mag.* 6.4, pp. 68–75.

Zhou, A. and Q. Zhang (2010). "A surrogate-assisted evolutionary algorithm for minimax optimization". In: *IEEE Congress on Evolutionary Computation*. IEEE, pp. 1–7.

Zhou, Z., Y. S. Ong, M. H. Nguyen, and D. Lim (2005). "A study on polynomial regression and gaussian process global surrogate model in hierarchical surrogate-assisted evolutionary algorithm". In: *IEEE Congress on Evolutionary Computation*. Vol. 3. IEEE, pp. 2832–2839.

A.2 Combining statistical learning with metaheuristics for the multi-depot vehicle routing problem with market segmentation

Laura Calvet[1], Albert Ferrer[2], M. Isabel Gomes[3], Angel A. Juan[1], David Masip[1]

1. Computer Science Department, IN3-Open University of Catalonia, Castelldefels, Spain
e-mail: {lcalvetl, ajuanp}@uoc.edu
2. Department of Mathematics, Technical University of Catalonia, Barcelona, Spain
e-mail: alberto.ferrer@upc.edu
3. Centro de Matemática e Aplicações, FCT, Universidade Nova de Lisboa, Lisbon, Portugal
e-mail: mirg@fct.unl.pt

Abstract

In real-life logistics and distribution activities it is usual to face situations in which the distribution of goods has to be made from multiple warehouses or depots to the final customers. This problem is known as the Multi-Depot Vehicle Routing Problem (MDVRP), and it typically includes two sequential and correlated stages: (a) the assignment map of customers to depots, and (b) the corresponding design of the distribution routes. Most of the existing work in the literature has focused on minimizing distance-based distribution costs while satisfying a number of capacity constraints. However, no attention has been given so far to potential variations in demands due to the fitness of the customer-depot mapping in the case of heterogeneous depots. In this paper, we consider this realistic version of the problem in which the depots are heterogeneous –in terms of their commercial offer– and customers show different willingness to consume depending on how well the assigned depot fits their preferences. Thus, we assume that different customer-depot assignment maps will lead to different customer-expenditure levels. As a consequence, market-segmentation strategies need to be considered in order to increase sales and total income while accounting for the distribution costs. To solve this extension of the MDVRP, we propose a hybrid approach that combines statistical learning techniques with a metaheuristic framework. First, a set of predictive models is generated from historical data. These statistical models allow estimating the demand of any customer depending on the assigned depot. Then, the estimated expenditure of each customer is included as part of an enriched objective function as a way to better guide the stochastic local search inside the metaheuristic framework. A set of computational experiments contribute to illustrate our approach and how the extended MDVRP considered here differs –in terms of the proposed solutions– from the traditional one.

Keywords: Multi-Depot Vehicle Routing Problem, market segmentation applications, hybrid algorithms, statistical learning

1. Introduction

In the distribution business, whenever a supplier operates from multiple warehouses or depots it needs to decide two things: (a) which set of customers will be served from each depot, i.e., the customer-depot assignment map; and (b) the vehicle routing plan for the given assignment map. This two-stage decision-making process is called the Multi-Depot Vehicle Routing Problem (MDVRP). During the last decades, researchers have extensively addressed different variants of this problem, among others those including heterogeneous fleets of vehicles, multiple products, simultaneous pick-up and delivery, etc. (Caceres et al., 2014; Montoya-Torres et al., 2015). The large majority of models aim at minimizing total distribution costs, which are often modeled by means of a distance-based cost function. Minimization of distribution costs has a major impact on the efficiency of any competitive shipping company. However, following the trend to consider richer and more realistic Vehicle Routing Problems (Ehmke et al., 2015; Barbucha, 2014; Taş et al., 2014), it should be noticed that these costs represent only half of the equation, i.e.: if a distribution company wants to maximize its benefits, it has also to account for the expected incomes associated with different customer-to-depot assignment plans. Thus, retail centers (depots) belonging to the same organization may offer different products, trade credit policies, or complementary services,

which often have a non-negligible impact on the customer's willingness to buy. Accordingly, under the existence of a diversity of depots and commercial offers, the customer-to-depot assignment process should not only consider distribution costs but also expected sales or total income.

In order to increase sales revenue, companies use market segmentation strategies that allow grouping customers according to their features (preferences, rent, age range, etc.). Ideally, each group has homogeneous features that allow the development of tailored strategies and actions oriented to increase the customer's willingness to buy, i.e., the fitness between his/her utility function and the commercial offer he/she is receiving. In this paper we address an extended version of the MDVRP that also includes market segmentation issues in order to maximize benefits (sales revenue minus distribution costs). Thus, in our model customer-to-depot assignation decisions are taken considering not only the traditional distance-based cost but also other customers' features in an attempt to increase the expected expenditure by providing a more adequate assignation. As a consequence of this, the assignment and routing solutions might be very different from the ones associated with the classical MDVRP. For instance, Figure 1 shows two different solutions, with the shape of each customer representing the shape of its best-fit depot. The one on the left only considers distribution costs (to be minimized), while the one on the right considers expected benefits (to be maximized), i.e.: not only distribution costs but also additional revenue due to a 'smarter' customer-to-depot assignment. Notice that in the right-hand solution each depot tends to deliver those customers that share a similar shape, unless they are too far away so that the increase in distribution costs overshadows the potential increase in revenue. In the illustrative example of Figure A.1, it is estimated that customer j will spend 20 monetary units when assigned to depot 2 (left-hand solution). On the other hand, if this same customer is assigned to depot 1 (right-hand solution), it is estimated that his/her willingness to spend will increase up to 30 monetary units. Therefore, assigning customer j to depot 1 instead of to its closes depot (depot 2) will pay off as far as the increase in transportation costs will not exceed the marginal income attained (10 monetary units in this case).

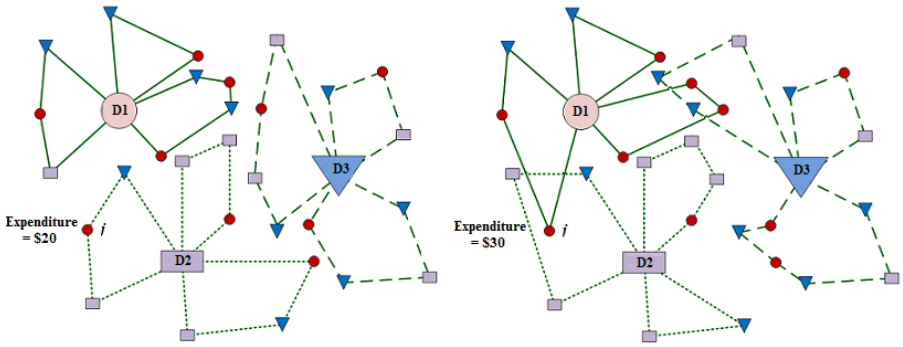


FIGURE A.1: Solutions for the classical MDVRP (left) and for the extended version (right)

Our solving approach is based on the combination of statistical predictive models with a metaheuristic framework. In short, the algorithm develops in two main steps. Firstly, supported by the company historical data concerning existent customers, new customers are assigned to depots. This step is preceded by a historical data analysis so that expected expenditure from new customers among depots is estimated throughout a multiple regression model. The regression model will capture the relationship between each customer's willingness to spend (response) as a function of several variables (predictors), including: the assigned depot as well as other customer's features (e.g.: preferences, rent, sex, age, etc.). In the second step, the routes associated to each customer-to-depot assignment map are built. Given the interdependency between both decisions (assignation and routing), our procedure is an iterative one. Different assignations are generated together with the routing decisions and the top best solutions will be saved and locally improved in the last step of the algorithm. The main contributions of our work are: (i) the description of an extended version of the MDVRP with heterogeneous depots, which can be considered a rich routing problem, (ii) the development of a methodology combining statistical learning and a metaheuristic for solving it, and (iii) an analysis of how the solutions found for the extended problem

differ from those for the classical one in terms of both expected benefits and distribution costs for a set of instances artificially generated.

The rest of the paper is organized as follows: Section 2 formally describes the well-known Multi-Depot Vehicle Routing Problem and presents the extended version with heterogeneous depots, while Section 3 reviews works addressing the classical version. Section 4 discusses the importance of considering market segmentation. Section 5 provides an overview on our solving approach, while Section 6 offers some low-level details. The computational experiments and a discussion of the results are presented in Section 7. Lastly, the main contributions of this work are highlighted in the Conclusion section.

2. Mathematical Formulation for the Multi-Depot Vehicle Routing Problem

The MDVRP may be formally described as an extension of the Capacitated Vehicle Routing Problem (CVRP) and it is defined as a complete directed graph $G = (V,\ E)$, where $V = \{V_d, V_c\}$ is the set of nodes including the depots, V_d, and the customers, V_c, and E is the set of edges or arcs connecting all nodes in V. Each customer i in V_c has a positive demand to be satisfied, q_i. Each edge in E has an associated cost $c_{i,j} > 0$ and distance $d_{i,j} > 0$ between customers i and j. The distance matrix $D := [d_{i,j}]$ and the cost matrix $C := [c_{i,j}]$ are square matrices of order $|V|$. Usually, both matrices are assumed to be symmetric (nevertheless, our approach could also be applied even in the case of non-symmetric distances or costs).

For the MDVRP, a solution is a customer-to-depot assignment map together with a set of routes covering all customers' demands. Each route starts at one depot in V_d, connects one or more customers in V_c, and ends at the same depot, without exceeding the capacity of the vehicle. The number of vehicles based at each depot may be fixed or unlimited. The former defines a harder problem, since it adds an additional constraint and there is also no guarantee that a feasible solution exists (Chao et al., 1993). The latter simplifies the modelling and solving.

As mentioned before, when adopting a marketing perspective, companies focus on market segmentation to group customers according to their features and preferences. Considering the heterogeneity of markets, segmentation attempts to divide customers into subsets that behave in a similar way. Our extension of the MDVRP aims at assigning customers to depots based not only on distribution costs but also on customers' features and preferences. The goal is then to optimize expected benefits by considering both distribution costs and expected incomes.

To formally describe the mathematical model for the MDVRP with heterogeneous depots, we will first introduce a model for the CVRP problem, which is a particular case of the MDVRP when $|V_d| = 1$, i.e., $V_d = \{0\}$, and a model for the classical MDVRP.

Mathematical Model for the MDVRP with One Depot (CVRP)

In graph theory, a finite path, ϕ, of length r is a sequence of $r + 1$ vertices, $\{\alpha_0,\ \alpha_1, \ldots,\ \alpha_r\}$, together with a sequence of r arcs, $\{\phi^1,\ \phi^2, \ldots,\ \phi^r\}$, such that

$$\phi^k = (\alpha_{k-1},\ \alpha_k),\ k = 1,\ 2, \ldots,\ r.$$

Sometimes we will denote a finite path, ϕ, in the form:

$$\phi :\ \alpha_0 \to \alpha_1 \to \alpha_2 \to \cdots \to \alpha_{r-1} \to \alpha_r.$$

The vertex α_0 is called the start vertex and the vertex α_r is called the end vertex of the path. Both of them are called terminal vertices of the path. The other vertices in the path are internal vertices. A finite cycle is a path such that the start vertex and the end vertex are the same. Note that the choice of the start vertex in a cycle is arbitrary. A path with no repeated vertices is called a simple path, and a cycle with no repeated vertices or arcs aside from the necessary repetition of the start and the end vertex is a simple cycle.

Definition A.2.1 *In our context, a route, ρ, of order r is a simple finite cycle of length $r + 2$ in which the start vertex and the end vertex is the depot node* 0*,*

$$\rho :\ 0 \to \alpha_1 \to \alpha_2 \to \cdots \to \alpha_{r-1} \to \alpha_r \to 0.$$

We denote, $\mathcal{R}$, the set of all routes of the complete directed graph G.

Notice that the cardinality of $\mathcal{R}$ is $|\mathcal{R}| = \sum_{k=1}^{n} P(n,\ k)$, where $P(n,\ k)$ represents the number of k-permutations of a set of n elements (or customers in our case). Notice that $|\mathcal{R}| = \sum_{k=1}^{n} P(n,\ k) \approx n!e$, where e represents the Euler's number, $e = \sum_{k=0}^{\infty} \frac{1}{k!}$.

Definition A.2.2 *Two routes are independent when they have no internal vertices in common, i.e., the only vertex in common is the depot node. A non-empty set of independents routes, $\mathcal{S} \subset \mathcal{R}$, is named a* **complete system of routes** *when every customer belongs to a route of $\mathcal{S}$. The set of all the complete system of routes of $\mathcal{R}$ is denoted by $\mathcal{CSR}$.*

Notice that from now, in order to simplify the notation, when we write $\alpha \in \rho$, with $\rho \in \mathcal{S}$, and $\mathcal{S} \in \mathcal{CSR}$, we want to indicate that α is a node of the route ρ.

Traditionally, the cost of a route, c_ρ, and its distance, d_ρ, have been modeled as

$$c_\rho := c_{\alpha_r,\alpha_0} + \sum_{k=1}^{r} c_{\alpha_{k-1},\ \alpha_k}, \quad d_\rho := d_{\alpha_r,\alpha_0} + \sum_{k=1}^{r} d_{\alpha_{k-1},\ \alpha_k}.$$

Then, the optimization problem to be solved consists in finding a complete system of routes, $\mathcal{S}$, minimizing the total cost, $c_T := \sum_{\rho\in\mathcal{S}} c_\rho$ subject to the following constraints: the total demand served in each route $\rho \in \mathcal{S}$ does not exceed a maximum constant demand (or vehicles capacity) Q_{max}, $\sum_{\alpha\in\rho} q_\alpha \le Q_{max}$, and the total distance of each route $\rho \in \mathcal{S}$ does not exceed a maximum constant distance D_{max}, $d_\rho \le D_{max}$. Therefore, the optimization problem is

$$\begin{array}{lll} \text{minimize} & c_T = \sum\limits_{\rho\in\mathcal{S}} c_\rho & \\ \text{subject to:} & \sum\limits_{\alpha\in\rho} q_\alpha \le Q_{max}, & \rho \in \mathcal{S} \\ & d_\rho \le D_{max}, & \rho \in \mathcal{S} \\ & \mathcal{S} \in \mathcal{CSR}. & \end{array} \tag{A.1}$$

Mathematical Model for the classical MDVRP

The extension to a MDVRP goes as follows: consider a complete directed graph $G = (V,\ E)$, where V is the disjoint union (also named a partition) of the set of nodes including the depots, V_d, and the set of nodes including customers V_c, $V := V_d \cup V_c$, and E is the set of edges connecting all nodes in V. Hereafter, $m := |V_d|$ will represent the number of depots. A feasible solution for the MDVRP is a partition of direct graphs $G_i = (V_i,\ E_i)$, $i = 1, \ldots, m$, obtained from G such that $V_i := \{0_i;\ v_1^i, \ldots,\ v_{m_i}^i\}$, for all $i = 1, \ldots, m$, with $0_i \in V_d$ and $v_j^i \in V_c$ for all $j = 1, \ldots, m_i$. Then, the optimization problem to solve consists in finding a family of complete system of routes, $\{\mathcal{S}_1, \ldots, \mathcal{S}_m\}$, minimizing the total cost, $c_T := \sum_{i=1}^{m} \sum_{\rho\in\mathcal{S}_i} c_\rho$ subject to the following constraints: the total demand served in each route $\rho \in \mathcal{S}_i$, $i = 1, \ldots, m$, does not exceed a maximum constant demand, Q_{max}, i.e., $\beta_\rho := \sum_{\alpha\in\rho} q_\alpha \le Q_{max}$, for all $\rho \in \mathcal{S}_i$, $i = 1, \ldots, m$, and the total distance of each route $\rho \in \mathcal{S}$ does not exceed a maximum constant distance D_{max}, i.e., for all $\rho \in \mathcal{S}_i$, $d_\rho \le D_{max}$, $i = 1, \ldots, m$. Therefore, the optimization problem is

$$\begin{array}{lll} \text{minimize} & c_T = \sum\limits_{i=1}^{m} \sum\limits_{\rho\in\mathcal{S}_i} c_\rho & \\ \text{subject to:} & \beta_\rho \le Q_{max}, & \rho \in \mathcal{S}_i,\ i = 1, \ldots, m, \\ & d_\rho \le D_{max}, & \rho \in \mathcal{S}_i,\ i = 1, \ldots, m, \\ & \mathcal{S}_i \in \mathcal{CSR}, & i = 1, \ldots, m. \end{array} \tag{A.2}$$

Mathematical Model for the MDVRP with heterogeneous depots

The heterogeneous version of the MDVRP analyzed in this paper does not assume depots are equal (homogeneous), which leads to consider customers' preferences. Then, demands will not be fixed parameters, but depend on the assignment map of customers to depots. Following a realistic approach, we assume demands are not known, but can be predicted relying on an historical database and information about new customers. In the heterogeneous case the assignation of the customers is not made in advance using the classical considerations of distance. Our procedure

takes into account the combination of statistical predictive models with a metaheuristic, so three main steps must be considered.

i) Analysis of the historical data so that expected expenditure from new customers among depots is estimated using a multiple regression model. The model captures the relationship between each customer's willingness to spend (response) as a function of several variables (predictors), which include the assigned depot as well as other customer's characteristics as preferences, rent, sex, age, and so on.

ii) Assignation of the new customers to the depots supported by the company historical data with respect to the existent customers.

iii) Routes are built, which are associated to each customer-to-depot assignment map.

Notice that revenue incomes are not considered in the model for the classical MDVRP because they do not depend on the assignation of customers to depots and, consequently, they are a constant value. On the other hand, given the interdependency between both assignation and routing, the procedure is an iterative one. Different assignations are generated (see Figure A.1) then, together with the routing decisions. The top best solutions will be saved and locally improved in the last step of the algorithm in order to maximize the total benefit, b_T, obtained from the difference between the total income, $i_T := \sum_{i=1}^{m} \left(\sum_{\rho \in \mathcal{S}_i} \beta_\rho \right)$ and the total cost $c_T := \sum_{i=1}^{m} \left(\sum_{\rho \in \mathcal{S}_i} c_\rho \right)$.

$$b_T := i_T - c_T = \sum_{i=1}^{m} \sum_{\rho \in \mathcal{S}_i} \left(\beta_\rho - c_\rho \right).$$

Thus, the optimization problem for the heterogeneous case can be described as

$$\begin{array}{lll} \text{maximize} & b_T := \sum\limits_{i=1}^{m} \sum\limits_{\rho \in \mathcal{S}_i} \left(\beta_\rho - c_\rho \right) & \\ \text{subject to:} & \beta_\rho \leq Q_{max}, & \rho \in \mathcal{S}_i,\ i = 1, \ldots, m, \\ & d_\rho \leq D_{max}, & \rho \in \mathcal{S}_i,\ i = 1, \ldots, m, \\ & \mathcal{S}_i \in \mathcal{CSR}, & i = 1, \ldots, m. \end{array} \tag{A.3}$$

3. Literature Review on the classical MDVRP

The MDVRP has received a considerable amount of attention in the recent literature (Montoya-Torres et al., 2015). Tillman (1969) is usually referred as the first paper to address this problem. It considers a version where customer demands follow specific probability distributions, which is solved with an extension of the well-known CWS heuristic (Clarke and Wright, 1964). Most works may be classified according to the proposed approach: exact methods and heuristics/metaheuristics methods. The main difference is that the former guarantee the optimality of the solution found, while the latter usually provide a high-quality solution faster. Currently, hybrid approaches have received more attention. Ceselli et al. (2009) is an example of work employing an exact methodology. The authors describe a version of the Multi-Depot Heterogeneous Vehicle Routing Problem with Time Windows (MDHVRPTW) including diverse constraints. A column generation algorithm, in which the pricing problem is a resource-constrained elementary shortest-path problem, is implemented to solve real instances. Another methodology to solve the MDHVRPTW is proposed in Bettinelli et al. (2011). It describes a branch-and-cut-and-price algorithm, and different pricing and cutting techniques. More recently, Contardo and Martinelli (2014) have formulated the MDVRP employing a vehicle-flow and a set-partitioning formulation.

A higher number of published works rely on heuristics-based methodologies. For instance, Cordeau et al. (1997) present a Tabu Search (TS) metaheuristic. In Salhi and Sari (1997), the authors propose a multi-level composite heuristic for addressing a MDVRP in which the vehicle fleet composition has to be determined. Nagy and Salhi (2005) consider the MDVRP with Pickups and Delivers. Several heuristics from the Vehicle Routing Problem (VRP) literature are adapted and some problem-specific are constructed. Metaheuristics are frequently implemented to solve real-size instances. The Simulated Annealing (SA) metaheuristic is chosen in Wu et al. (2002) for solving the Multi-Depot Location-Routing Problem. Polacek et al. (2004) employ the Variable Neighborhood Search (VNS) metaheuristic for addressing the MDVRP with Time Windows (MDVRPTW). The MDVRP with a heterogeneous fleet of vehicles is faced in Salhi et al.

(2014), where an algorithm also based on the VNS metaheuristic is designed. Pisinger and Ropke (2007) tackle different variants of the VRP, including the MDVRP, by transforming them into rich pickup and delivery models and developing an Adaptive Large Neighborhood Search methodology. A Genetic Algorithm (GA) is constructed in Ombuki-Berman and Hanshar (2009). Another population-based metaheuristic, the Path Relinking, is presented in Rahimi-Vahed et al. (2013).

Regarding hybrid algorithms, Ho et al. (2008) introduce an algorithm relying on a GA. The initialization procedure consists in a distance-based grouping, the CWS heuristic is employed for routing, and the Nearest Neighbor Heuristic (NNH) for scheduling (i.e., sequencing each route in every depot). Another hybrid GA is developed in Vidal et al. (2013) for addressing several rich VRPs, including the MDVRPTW. It has diversity management mechanisms, and employs geometric and structural problem decompositions for large instances. Mirabi et al. (2010) describe a methodology combining a constructive heuristic search and improvement techniques. First, the nearest depot method, the CWS heuristic and the NNH are implemented for grouping, routing, and scheduling, respectively. The resulting solutions are improved by means of a deterministic, stochastic, or the SA metaheuristic. Yu et al. (2011) construct an algorithm based on the Ant Colony metaheuristic, applying a coarse-grain parallel strategy, an ant-weight strategy and mutation operation. Cordeau and Maischberger (2012) design a parallel Iterated Tabu Search heuristic which introduces the TS heuristic into the Iterated Local Search (ILS) framework, in order to ensure a broad exploration of the search space. The Particle Swarm Optimization (PSO) metaheuristic is proposed in Geetha et al. (2012). It generates initial particles with the k-means algorithm and the NNH. Lahrichi et al. (2012) present a multi-thread cooperative search method called the Integrative Cooperative Search for multi-attribute combinatorial optimization problems. In Juan et al. (2015c), the authors combine an ILS metaheuristic with biased-randomization techniques to solve the MDVRP. The same metaheuristic framework is proposed in Li et al. (2015). In this case, an adaptive neighborhood selection mechanism is integrated for the MDVRP with simultaneous deliveries and pickups. Luo and Chen (2014b) develop an improved Shuffled Frog Leaping Algorithm (SFLA) and its multi-phase model for the MDVRP and the MDVRPTW. In order to improve the efficiency of the metaheuristic, a Power Law Extremal Optimization Neighborhood Search is used. The same problems are addressed in Luo and Chen (2014a), where a multi-phase modified SFLA is applied. It implements the k-means algorithm and presents cluster and global optimization procedures.

4. Importance of considering Market Segmentation

In a global and dynamic world, companies have to compete in order to build profitable and long-lived relationships with customers. Analyzing customer needs and desires, capabilities, social values, and objectives of a specific company –as well as how these interrelate– is a crucial area in business intelligence. During many decades mass market-based strategies had prevailed, which make profit from economies of scale, providing homogeneous goods and services for a vast number of customers. Technological developments and flexible manufacturing systems have boosted the customization of goods and services according to customer preferences (Datta, 1996; Liu et al., 2012b). Market segmentation is a key concept in this new approach.

Considering the heterogeneity of markets, segmentation attempts to divide customers into subsets that behave in the same way or have similar needs (Bennett, 1995). As a result, a better understanding of customer requirements is obtained, which may assist in the developing of marketing strategies as well as in the efficient allocation of resources among markets and products (Wind, 1978). According to Foedermayr and Diamantopoulos (2008), the segmentation process includes the following stages (Figure A.2):

1. Market definition: The scope of the concept of market for a company is chosen. It should be broad enough to cover as many potential customers as possible, but also manageable.

2. Selection of segmentation variables or bases: These bases should be capable of diminishing the market heterogeneity and explaining why customers have different requirements and/or do not respond similarly to marketing campaigns. From the point of view of the company, they should be easy to obtain or infer in terms of cost and time, among others. The most popular are classified into the following groups (Kotler and Armstrong, 2011): (i) geographic bases (e.g., location); (ii) demographic bases (e.g., age, occupation, and education level);

(iii) behavioral bases (e.g., purchase occasion, degree of usage, and degree of loyalty); and (iv) psychographic bases (customer activities and opinions).

3. Decision on segmentation method: A-priori versus post-hoc methods, and descriptive versus predictive methods, are the criteria most commonly employed to classify segmentation methods (Foedermayr and Diamantopoulos, 2008). A-priori methods are based on intuitions and prior experience, and/or secondary data. While in post-hoc methods the data analysis is what leads to the segments. In descriptive methods, no distinction is made between dependent and independent variables. The focus is on exploring the relation between the units of analysis and the variables. In contrast, predictive methods link a dependent variable (e.g., degree of loyalty) to a set of independent variables, and use this set to segment. There are plenty of techniques for segmentation, which includes: cross tabulation analysis, RFM analysis, k-means clustering, hierarchical clustering, self-organizing map (SOM), automatic interaction detection, classification and regression trees, logistic regression, support vector machine, linear regression, clusterwise regression, neural networks, finite mixture model, and metaheuristics, among others. For instance, McCarty and Hastak (2007) investigate RFM, decision trees, and logistic regression. Vellido et al. (1999) present a strategy combining SOM and factor analysis before clustering. Another two-stage approach involving SOM is detailed in Kuo et al. (2014). These authors apply SOM to determine the number of clusters and the starting point, and the k-means algorithm to find the final solution. Huang et al. (2007) employ a support vector clustering algorithm. Fish et al. (1995) analyze the performance of artificial neural networks, in comparison with those of discriminant analysis and logistic regression. A case-based reasoning system is described in Chen et al. (2010). It implements GAs for selecting variables and instances.

4. Formation of market segments: The method selected in the previous step is applied to obtain a set of segments.

5. Profiling, evaluation, and final selection of target segments: A detailed analysis of the resulting segments and a selection of them are performed. There are several criteria to evaluate market segments. Smith (1956), considered the first work to tackle this issue, highlights the characteristics of *identifiability*, which means that customers in a segment should have a similar profile, allowing for their identification, and *responsiveness*, i.e., customers in a segment should similarly respond to a marketing strategy. DeSarbo and DeSarbo (2007) gather the main criteria that have been proposed in the literature. Some examples are: *reachability*, *feasibility*, *profitability*, and *stability*.

6. Implementation: The next step is to translate the results of the previous work into specific strategies. This step involves decisions that depend on a large number of factors as relevant as company resources and ethics.

7. Segmentation strategy evaluation: Sales, profit, company expansion, reputation, and customer satisfaction may be used to evaluate a strategy. Although these steps could be sequentially followed, all are interconnected. Therefore, it is recommendable to allow the possibility to repeat previous steps in order to reconsider some selections.

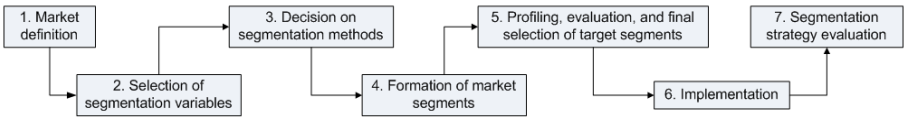


FIGURE A.2: Scheme of the segmentation process.

As it has been shown, marketing segmentation has been extensively studied. It is an important topic of research due to its potential applications. New lines of research emerge from the development of data techniques, the gathering of empirical evidences, and the publication of new marketing theories, among others. Many challenges still remain to be faced such as reducing the gap between academic research and practitioner needs, studying implementation issues.

5. Overview of Our Approach

The MDVRP includes two sequential and correlated stages: (a) the assignment map of customers to depots; and (b) the corresponding design of distribution routes to satisfy all customers' demands. In order to assign customers, we take into account the heterogeneity of the depots. It can be considered a realistic approach, since depots belonging to the same organization usually have different characteristics related to products, trade credit policies, and complementary services, among others. The diversity of depots leads to consider customer preferences. Specifically, the willingness to consume (or expenditure) of each customer depends on how well the assigned depot fits his/her preferences. Market segmentation techniques are applied to identify subsets of customers with similar profiles and assign them to the particular depot that better fits their preferences, considering the restrictions of the problem. Accordingly, we propose to study the relationship between expenditure and customers' features from data of existent customers by employing statistical learning methodologies (e.g., prediction techniques). It will enable the assignation of new customers in such a way that the expected benefits (expected incomes minus distribution costs) is maximized. The phases of our approach are represented in Figure A.3 and described next:

1. Data collection. Our approach requires several inputs: database of historical sales, description of new customers, location of depots, vehicle maximum capacity, number of available vehicles at each depot, and maximum distribution costs per route. The sales database includes the following information for each existent customer: personal features, geographical location, expenditure level, and depot to which he/she has been assigned (randomly or according to a metric not related to personal features such as distribution costs). The description of new customers gathers personal features and geographical locations. This information may be easily obtained, for instance, in e-commerce environments, where customers have to register and provide personal data before buying. After processing and analyzing this data, a company may assign a new client by redirecting him/her to a specific directory/website and offering goods from a given depot. Regarding the information of both existent and new customers, an initial selection of variables has to be performed by assessing which ones may be valuable. Besides explaining the differences of expenditures among depots, they should be easy to obtain, estimate or compute, and store.

2. Statistical learning. Given the database of existent customers, a statistical model exploring the relationship between customers' features and expenditure is performed for each group of customers assigned to a specific depot. Considering several groups, we allow the existence of a different trend in each one. A high number of methodologies are available to carry out regression analysis (Hastie et al., 2009; Lantz, 2013). Probably, the most applied is Linear Regression (Montgomery et al., 2012), which is easy to understand and interpret, highly relevant in the marketing literature, and has associated a relatively low risk of overfitting (i.e., the model describing noise). Neural Networks represent a popular alternative capable of capturing non-linear relationships. However, they are computationally more intensive, may overfit/underfit data more easily and are difficult to interpret. Support Vector Machines constitute another powerful black box approach, which is more robust and less prone to overfitting than Neural Networks. Its main disadvantage is that requires testing several combinations of kernels and model parameters. Model Trees combine Decision Trees with modeling of numeric data. It results in an approach that may fit some types of data better than linear regression and perform automatic feature selection. On the other hand, it may be difficult to determine the overall net effect of individual variables on the response.

3. Prediction of expenditure for new customers. Once a methodology has been selected and the different functions have been fitted, the expenditure is predicted for each new customer given his/her features if assigned to each depot. Here, it is assumed that the sample (set of existent customers) is representative of the population (market).

4. Assignment of customers to depots. In order to perform an efficient and feasible assignation, it is necessary not only to consider the predicted expenditure but also the distribution costs, the maximum number of vehicles per depot, and their capacity. Taking a decision for each customer individually may provide non-feasible and poor-quality solutions. Consequently, we present a global and iterative strategy where customers are selected one at a time to be assigned to a specific depot. It prioritizes the assignments of those customers that have

associated a relatively high expected benefits only for a particular depot, and is based on the procedure developed in Juan et al. (2015c). In particular, the following steps are proposed:

- For each depot k and customer i,
  - Compute the expected benefits μ_i^k as the difference between the predicted expenditure p_i^k and the distribution costs c_i^k (computed as the cost of moving from k to i).
  - Compute the difference between the expected benefits of assigning i to k and the maximum expected benefits of assigning i to a depot l other than k, i.e.:

$$s_i^k = \mu_i^k - max_{l \in V_d \setminus \{k\}} \mu_i^l \quad \forall i \in V_c, \ \forall k \in V_d$$

    We refer to this measure as "marginal savings". Accordingly, s_i^k will be high in the case customer i reports relevant expected benefits only if assigned to k, low (in absolute terms) if the expected benefits are similar for k and at least one other depot, presenting both depots the highest expected benefits, and very low (negative) when there is at least one depot where the expected benefits are larger than those estimated for k.
- For each depot k, create a priority list of customers and sort it in descending order according to the marginal savings s_i^k.
- Create a list of unassigned customers. Then, select a depot and choose the next customer to assign from its priority list. Update the list of unassigned customers and repeat these steps while there are unassigned customers. Different policies may be applied to determine which depot selects the next customer, as: (i) allowing the depot with the highest remaining capacity to choose, (ii) using a round robin-based criterion, or (iii) selecting it randomly.

5. Routing. Having an assignment map, the MDVRP can be solved as a set of independent CVRPs. However, the most important challenge when addressing a MDVRP instance is the interrelation between assignation and routing. Therefore, algorithms are required to take the decisions associated to both phases 'simultaneously'. Thus, instead of finding an optimal or near-optimal solution for the customer-to-depot assignment phase and then use this unique solution as a starting point to solve the routing phase, an iteration process combines 'good' and fast computed solutions for the first stage with 'good' and fast computed solutions for the second one in order to find a near-optimal solution for the overall problem.

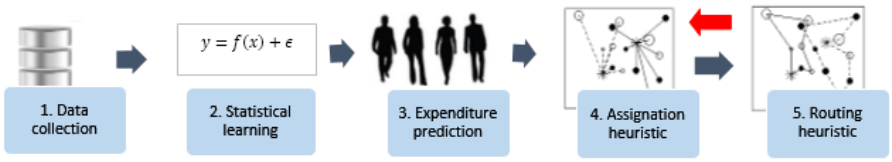


FIGURE A.3: The proposed approach.

Note that our approach will be appropriate as long as the existent customers had been assigned randomly or based on a variable not related to personal features. If regression functions were estimated again after implementing this procedure (replacing existent customers by the new ones), the predictive model could be not valid anymore, since the groups of customers assigned to each depot may not be representative of all potential customers. At this point, a description of each resulting group may be performed. Accordingly, a new customer would be assigned to the closest group (considering standardized data, the Euclidean distance, and an average profile per group, for instance).

In the described approach, the statistical learning techniques and the metaheuristic are sequentially employed. There are other realistic versions of the problem that may be addressed by adapting our approach to integrate the statistical learning techniques inside the metaheuristic. For instance, consider a dynamic scenario in which the willingness of customers to spend varies as

new customers are assigned to each depot (e.g., due to the decrease in the service's quality or in the number of available offers). In this case, the learning mechanism would iteratively run throughout the searching process in order to update each customer's willingness to spend after each assignment.

6. Detailed Algorithm

This section describes some low-level details of the proposed approach. Figure A.4 summarizes it highlighting the main differences between the classical version of the problem and the proposed one.

Since the phase of data collection is company-specific, we will assume it has already been done. The second and the third phases are related to the development and use of predictive statistical learning models. First, the database of existent customers is split into two subsets: a training set, which will be used to build the models, and a test set, to assess their performance. These subsets are generated by means of random sampling: 75% of customers are assigned to the training set and 25% to the test set. Having different alternatives to explore the relationship between expenditure and customers' features, in our experiments (described later in this paper) we make use of three well-known methodologies: Multiple Linear Regression (MLR), Multi-layer Feedforward Network (MFN), and Model Tree.

- Regarding Multiple Linear Regression, given a database of customers with m features and $|V_d|$ depots, the models proposed may be described as follows:

$$Exp_i = \beta_0^j + \beta_1^j \cdot f_{1i} + \beta_2^j \cdot f_{2i} + ... + \beta_m^j \cdot f_{mi} + \epsilon_i \quad \forall i \in V_c^j,\ \forall j \in V_d$$

  where f_{1i},...,f_{mi} represent the features of customer i, β_0^j,...,β_m^j are the parameters of the model, Exp_i and ϵ_i denote the expenditure and an error term for customer i, and V_c^j is the set of customers assigned to depot j. The ordinary least squares method is applied to estimate the parameters, and the stepwise regression approach with a bidirectional elimination procedure is chosen to perform the variable selection.

- Regarding the Multi-layer Feedforward Network with one hidden layer, the generated models are:

$$Z_{li} = \sigma(\beta_0^{jl} + \beta_1^{jl} \cdot f_{1i} + \beta_2^{jl} \cdot f_{2i} + ... + \beta_m^{jl} \cdot f_{mi}) \quad \forall i \in V_c^j,\ \forall j \in V_d,\ l = 1, ..., p$$

$$Exp_i = \alpha_0^j + \alpha_1^j \cdot Z_{1i} + ... + \alpha_p^j \cdot Z_{pi} \quad \forall i \in V_c^j,\ \forall j \in V_d$$

  where σ is the sigmoid function and p the number of hidden units. The value of p (4, 5, 6, 7, or 8) and the decay value for regularization (0.2, 0.3, 0.4, 0.5 or 0.6) are set using 10-fold cross validation based on the metric R^2 (Kuhn, 2008). The back propagation method is employed to estimate the parameters.

- The algorithm selected to implement a model tree is the standard M5P (Wang and Witten, 1996). Basically, it builds a decision-tree induction algorithm relying on a splitting criterion that minimizes the intra-subset variation in the class values down each branch. The pruning of the tree is performed back from each leaf. Instead of a constant value, the final solution for each leaf is a linear regression model considering the variables participating in decisions.

Different criteria can be employed to select one of the former statistical learning methodologies. The most common criteria are related to performance, easiness to apply and understand, required time, or any combination of the aforementioned properties. Considering the first one, we compute the Mean Squared Error (MSE) for each model (the number of models is the number of depots multiplied by the number of methodologies tested) using the same problem instance. The Total MSE (TMSE) is computed by aggregating the values of the models corresponding to the same methodology. In mathematical terms:

$$MSE^{aj} = \frac{1}{|V_c^j|} \sum_{\forall i \in V_c^j} (\widehat{Exp_i^a} - Exp_i)^2 \quad \forall a = 1, ..., o \quad \forall j \in V_d$$

$$TMSE^a = \sum_{j=1}^{|V_d|} MSE^{aj} \quad \forall a = 1, ..., o$$

where a represents the methodology assessed, and $\widehat{Exp}_i^a$ refers to the predicted expenditure for customer i employing the methodology a. In our experiments, for each instance we always select the methodology associated with the lowest $TMSE$. Thus, during the third phase, the expenditure that each new customer would make if he/she was assigned to each one of the depots is predicted using the selected methodology and the customer's features.

For the assignation and the routing phases, an existing methodology described in Juan et al. (2015c) has been adapted. The authors propose an efficient algorithm based on an ILS metaheuristic framework (Lourenço et al., 2010), which is a popular choice for solving routing problems (see Cattaruzza et al., 2014). This metaheuristic guides the search by interspersing exploration and intensification movements. Firstly, an initial solution is generated assigning customers to depots according to the marginal savings (only the distribution costs are considered) and designing the routes by implementing the classical CWS heuristic (Clarke and Wright, 1964). Afterwards, an iterative procedure is started in which the base solution (the initial solution in the first iteration) is perturbed. If the new solution is better than the base solution, then the latter is replaced. In case no improvement is achieved, a Demon-based acceptance criterion (Talbi, 2009) is considered to avoid entrapment at local optimum. It allows movements that deteriorate the base solution with a higher frequency at the beginning, when a global search is required, and restricts them as the execution proceeds. These steps are repeated until a termination condition is met. Finally, the top best solutions are improved by means of a post optimization process, and the best one is returned. The described algorithm includes Biased Randomization techniques to further diversify the search (Juan et al., 2009c). These techniques are introduced in traditionally deterministic steps in order to add biased randomization, which favors the generation of high-quality alternatives. In particular, they are implemented both in the assignation phase, to randomize the sorted priority list of customers of each depot in such a way that the reasoning behind the sorting is not erased but many orderings are provided, and in the routing phase, where the CWS heuristic is randomized.

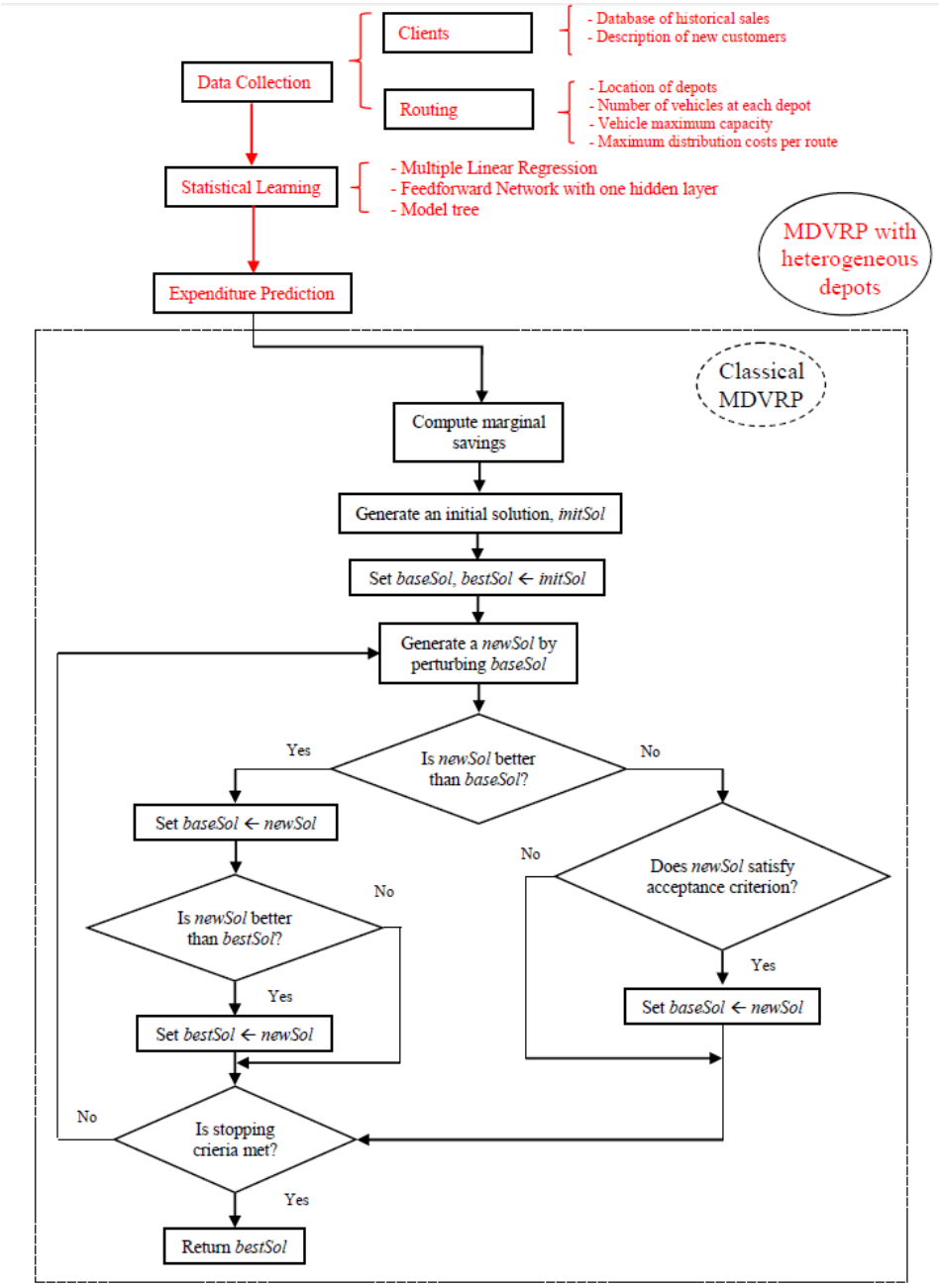


FIGURE A.4: Flow chart of our approach for solving the MDVRP with heterogeneous depots.

7. Numerical Experiments

An algorithm based on the described approach has been implemented and employed to solve a number of generated instances. The computational experiments compare the results of our approach for the analyzed version of the MDVRP and for the classical version (i.e., the one assuming homogeneous depots). This section provides the description of the instances and the tests carried out, as well as the numerical results and their analysis.

Set of instances

A total of 15 instances have been generated. Each of them consists in three datasets: the first two gather data concerning existent and new customers, respectively, and the third includes depots'

locations and information related to restrictions. Regarding data of existent customers, four variables have been created: age (a discrete variable following a Uniform distribution with parameters 16 and 80), sex (a categorical variable with two equally probable values), estimated income (it follows a Normal distribution with a mean of 1500 and standard deviation of 300), and preferred article (a categorical variable including four equally probable values). Initially, each customer has been assigned to his/her closest depot, while the expenditure level has been determined by a given function that depends on the depot, the aforementioned variables and a white noise term. For a total of 100 new customers, the variables age, sex, estimated income and preferred article have been generated using the same distributions. Customers' and depots' locations have been randomly generated in a square of 100 x 100. In order to simplify the instances' generation, Euclidean distances are employed as distribution costs. Different values have been chosen for the number of depots, existent customers and vehicles, the maximum cost per route and vehicles' capacity. This information is shown in Table A.1.

Instance	Numb. depots	Numb. existent cust.	Numb. vehicles	Vehicle capacity	Max. cost
1	3	300	3	250	200
2	3	300	3	225	200
3	3	300	3	225	150
4	3	300	3	225	200
5	3	300	3	200	150
6	3	400	3	350	225
7	3	400	3	300	200
8	3	400	3	200	175
9	5	400	4	325	175
10	5	400	4	200	150
11	5	400	4	275	175
12	5	400	4	275	150
13	5	400	4	225	200
14	5	400	4	175	125
15	5	400	4	250	175

TABLE A.1: Description of the generated instances.

Test

Each instance has been adapted by modifying the expenditure of existent customers to analyze the following scenarios: (1) low ratio (LR), the average ratio between average expenditure of existent customers and average distribution costs is similar; (2) medium ratio (MR), average expenditure is relatively higher than average distribution costs; and (3) high ratio (HR), average expenditure is much higher than average distribution costs. The target ratio has been reached multiplying expenditures by a coefficient. The resulting instances are available from the authors upon request. The analysis of these scenarios will allow us to compare the expected benefits (expected incomes, defined as the sum of predicted expenditures, minus distribution costs) associated to solutions considering only distribution costs and those taking into account also customer preferences (predicted expenditure), thus exploring the consequences of having different weights of expenditure in the objective solution. For the first scenario, it is expected that the gap between distribution costs will be low (i.e., solutions are expected to be relatively similar). Likewise, it is expected that this gap will be higher as the ratio increases. Similarly, it is also expected that the higher the ratio, the higher the gap between the expected benefits of the solutions. The code has been implemented with Java and R - version 2.15.0 (Team, 2008) (packages: caret, MASS, nnet, and RWeka). A standard personal computer, Intel QuadCore i5 CPU at 3.2 GHz and 4 GB RAM with Windows XP, has been used to perform all tests. The ILS process runs for 4,000 iterations, and all executions are solved for 10 different seeds. Only the best values obtained after the 10 runs are reported.

Results and analysis

The results of the experiments carried out are summarized in Figures A.5 and A.6. The boxplots in the first figure show the expected benefits per scenario and version of the problem: considering heterogeneous depots (rich) and assuming homogeneous ones (traditional). Even if the medians associated to each ratio level do not differ significantly, the third and second quartile values do present a higher value for the extended version of the problem. This behavior is caused by the long right tails of the corresponding distributions, which indicate that for some instances the rich version results in better solutions in terms of expected benefits. The second figure displays the variables in which expected benefits are decomposed per scenario and considering the rich version. We observe that differences of expected benefits between scenarios are mainly due to differences between expected incomes.

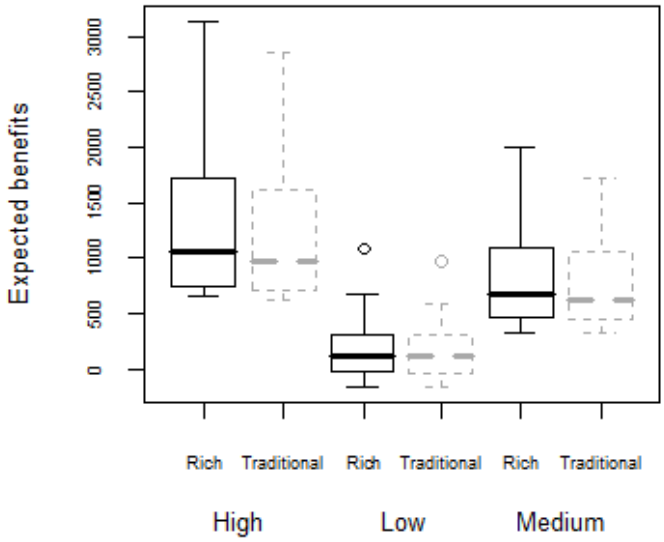


FIGURE A.5: Boxplot of the expected benefits for each scenario and version of the problem.

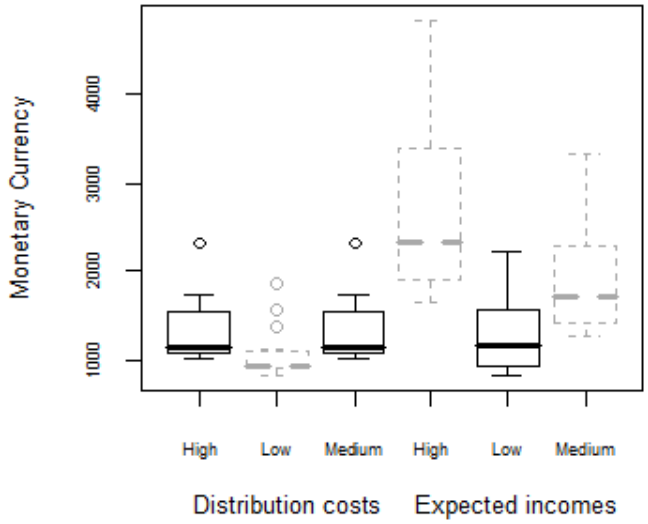


FIGURE A.6: Boxplot of the distribution costs and the expected incomes for the rich version of the problem.

Tables A.2, A.3 and A.4, provide a detailed description of the results. The information gathered in the tables is the following: instance name; methodology selected for prediction; distribution costs, expected incomes, expected benefits and time associated to the best solution found considering only distribution costs (classical MDVRP) and to the best solution found when maximizing expected benefit (MDVRP with heterogeneous depots); and gaps between distribution costs, expected incomes and expected benefits of both solutions. The average of each gap is also shown.

Given the flexibility of Feedforward Neural Networks to model relationships between variables, and despite the basic topology and parameter fine-tuning, and the medium size of the training set, they have been selected to solve more than half of the instances (57.8%). Multiple Linear Regression has provided the best TMSE in a high number of cases (31.1%). Although less frequently, the algorithm M5P has also been used in some instances (11.1%). Being an experiment for illustrative purposes, we show that different methodologies with particular strengths may be easily applied, but we do not aim to perform a comprehensive comparison among them.

The gaps related to the distribution costs and the expected incomes are strictly positive except in one case. It confirms the trade-off decision-makers face between both measures; that is to say, higher distribution costs are required to obtain an increase in expected incomes. Regarding the gap of expected benefits, it is strictly positive for all instances except for two where both solutions are equal. Therefore, attempting to achieve the highest benefits studying only distribution costs in instances with heterogeneous depots results in sub-optimal solutions. As expected, all average gaps increase with the ratio, i.e., the difference between solutions (in terms of distribution costs, expected incomes or expected benefits) is positively correlated to the average expenditure for fixed average distribution costs. However, this rule does not apply for all cases. In some of them, despite the fact that the gap of expected incomes increases, so does the gap of distribution costs. As a consequence, the gap of expected benefit may be reduced.

		Traditional (1)				Rich(2)				Gaps(2-1)		
Inst.	Meth.	Dist. cost	Exp. inc.	Exp. ben.	Time	Dist. cost	Exp. inc.	Exp. ben.	Time	Dist. cost	Exp. inc.	Exp. ben.
p01.1	MLR	898.6	961	62.4	82	930.6	1006	75.4	123	31.9	45.0	13.1
p02.1	M5P	834.3	943	108.7	112	834.5	947	112.6	335	0.1	4.0	3.9
p03.1	MFN	944.0	911	-33.0	143	964.4	939	-25.4	159	20.4	28.0	7.6
p04.1	MFN	891.8	852	-39.8	79	923.4	884	-39.4	165	31.6	32.0	0.4
p05.1	MFN	909.7	824	-85.7	189	914.4	829	-85.4	66	4.8	5.0	0.2
p06.1	MFN	868.5	1425	556.5	655	870.2	1429	558.8	613	1.7	4.0	2.3
p07.1	MFN	923.4	1073	149.6	103	925.7	1093	167.3	383	2.3	20.0	17.7
p08.1	M5P	898.2	867	-31.2	105	900.9	872	-28.9	122	2.7	5.0	2.3
p09.1	MLR	1039.2	2008	968.8	91	1127.5	2218	1090.5	33	88.3	210.0	121.7
p10.1	MFN	1029.6	1404	374.4	63	1062.5	1462	399.5	40	32.9	58.0	25.1
p11.1	MLR	880.7	1469	588.3	47	939.1	1609	669.9	464	58.4	140.0	81.6
p12.1	MFN	1858.4	1699	-159.4	108	1864.2	1709	-155.2	328	5.8	10.0	4.2
p13.1	MLR	1428.3	1495	66.7	437	1568.0	1691	123.0	144	139.6	196.0	56.4
p14.1	MFN	930.0	1163	233.0	43	930.0	1163	233.0	40	0.0	0.0	0.0
p15.1	M5P	1268.1	1401	132.9	374	1375.0	1512	137.0	59	107.0	111.0	4.0
								Average		35.2	57.9	22.7

TABLE A.2: Results obtained for 15 instances: scenario characterized by a low ratio.

8. Conclusions

This paper addresses an extension of the Multi-Depot Vehicle Routing Problem (MDVRP) in which heterogeneous depots are considered. The resolution of the classical MDVRP has two sequential and interrelated stages: (a) the assignment of customers to depots, and (b) the corresponding design of distribution routes. Typically, the assignment map is generated by minimizing the total distance, which is intended to lead to the minimization of distribution costs. Implementing this approach, researchers assume that depots are homogeneous. However, this is an unrealistic assumption since several factors may result in differences between depots from a particular organization. We propose to take into account the existence of heterogeneous depots, which allows the consideration of customers' preferences. The customers' willingness to consume is affected by how well the assigned depot fits their preferences. Thus, the main contribution of this work

		Traditional (1)				Rich(2)				Gaps(2-1)		
Inst.	Meth.	Dist. cost	Exp. inc.	Exp. ben.	Time	Dist. cost	Exp. inc.	Exp. ben.	Time	Dist. cost	Exp. inc.	Exp. ben.
p01.2	MLR	925.3	1383	457.7	277	978.0	1483	505.0	173	52.7	100.0	47.3
p02.2	MLR	901.2	1334	432.8	301	921.9	1385	463.1	254	20.7	51.0	30.3
p03.2	MLR	959.3	1405	445.7	134	979.1	1438	458.9	89	19.8	33.0	13.2
p04.2	MFN	942.5	1280	337.5	124	947.8	1292	344.3	101	5.3	12.0	6.7
p05.2	MFN	919.0	1264	345.0	51	921.3	1269	347.8	221	2.3	5.0	2.7
p06.2	MFN	945.6	2103	1157.4	106	948.6	2122	1173.4	327	3.1	19.0	15.9
p07.2	MFN	962.8	1581	618.2	394	992.3	1617	624.7	139	29.5	36.0	6.5
p08.2	MFN	969.9	1302	332.1	300	969.9	1302	332.1	296	0.0	0.0	0.0
p09.2	MFN	1169.6	2897	1727.4	36	1336.1	3335	1998.9	173	166.5	438.0	271.5
p10.2	MFN	1165.1	2109	943.9	161	1222.9	2222	999.1	97	57.8	113.0	55.2
p11.2	MLR	1001.8	2212	1210.2	80	1054.4	2288	1233.7	253	52.5	76.0	23.5
p12.2	MFN	1050.0	2571	1521.0	75	1070.5	2620	1549.5	41	20.6	49.0	28.4
p13.2	MLR	1633.4	2178	544.6	106	1778.2	2446	667.8	270	144.8	268.0	123.2
p14.2	MFN	1020.2	1703	682.8	63	1026.8	1717	690.2	67	6.6	14.0	7.4
p15.2	M5P	1419.6	2090	670.4	69	1560.2	2257	696.8	106	140.5	167.0	26.5
								Average		48.2	92.1	43.9

TABLE A.3: Results obtained for 15 instances: scenario characterized by a medium ratio.

		Traditional (1)				Rich(2)				Gaps(2-1)		
Inst.	Meth.	Dist. cost	Exp. inc.	Exp. ben.	Time	Dist. cost	Exp. inc.	Exp. ben.	Time	Dist. cost	Exp. inc.	Exp. ben.
p01.3	MLR	1060.3	1930	869.7	199	1153.7	2132	978.3	42	93.4	202.0	108.6
p02.3	M5P	1070.7	1803	732.3	253	1097.0	1864	767.0	174	26.3	61.0	34.7
p03.3	MFN	1042.7	1864	821.3	23	1067.1	1923	855.9	162	24.4	59.0	34.6
p04.3	MFN	1043.2	1701	657.8	54	1080.5	1755	674.5	393	37.2	54.0	16.8
p05.3	MFN	994.0	1621	627.0	174	1011.0	1657	646.0	68	17.0	36.0	19.0
p06.3	MFN	1068.1	2856	1787.9	109	1102.7	2906	1803.3	208	34.6	50.0	15.4
p07.3	MFN	1064.1	2115	1050.9	152	1081.2	2139	1057.8	71	17.1	24.0	6.9
p08.3	M5P	1069.6	1741	671.5	32	1069.6	1741	671.5	261	0.0	0.0	0.0
p09.3	MLR	1420.5	4269	2848.5	37	1690.6	4825	3134.4	138	270.1	556.0	285.9
p10.3	MFN	1434.8	2913	1478.2	113	1734.8	3396	1661.2	33	299.9	483.0	183.1
p11.3	MLR	1238.0	3020	1782.0	25	1486.3	3407	1920.7	265	248.3	387.0	138.7
p12.3	MFN	1195.7	3385	2189.3	37	1216.1	3452	2235.9	125	20.3	67.0	46.7
p13.3	MLR	1843.3	2801	957.7	79	2321.4	3387	1065.6	101	478.1	586.0	107.9
p14.3	MFN	1198.9	2297	1098.1	17	1251.0	2351	1100.0	23	52.1	54.0	1.9
p15.3	M5P	1416.0	2086	670.0	164	1595.5	2311	715.6	210	179.5	225.0	45.5
								Average		119.9	189.6	69.7

TABLE A.4: Results obtained for 15 instances: scenario characterized by a high ratio.

is the development of a simple yet comprehensive metaheuristic-based approach including market segmentation issues in order to maximize expected benefits (expected sales incomes minus distribution costs).

The proposed methodology consists of five steps: (i) data collection, in which information basically related to existent customers that have been already served and new customers is gathered; (ii) statistical learning, where the relationship between customers' features and expenditure for different depots is studied employing existent customer data; (iii) expenditure prediction for new customers; (iv) assignment of new customers; and (v) routing. A set of computational experiments has been carried out in order to illustrate our methodology. A total of 15 instances have been artificially generated and analyzed considering three scenarios, which vary in the weight of the expenditure of existent customers. It has been shown how our approach differs from an approach based only on minimizing distribution costs when solving instances with heterogeneous depots. Our experiment also allows quantifying how the performance gap between both approaches increases as the weight of the expenditures is incremented.

Acknowledgments

This work has been partially supported by the Spanish Ministry of Economy and Competitiveness (TRA2013-48180-C3-P, TRA2015-71883-REDT, MTM2014-59179-C2-01-P), FEDER

of EU, the Dept. of Universities, Research, and Information Society of the Catalan Government (2014-CTP-00001), and the Portuguese Foundation for Science and Technology (UID/MAT/00297/2013).

References

Barbucha, D. (2014). "A cooperative population learning algorithm for vehicle routing problem with time windows". In: *Neurocomputing* 146, pp. 210–229.

Bennett, P.D. (1995). *Dictionary of marketing terms*. 2nd. McGraw-Hill Contemporary.

Bettinelli, A., A. Ceselli, and G. Righini (2011). "A branch-and-cut-and-price algorithm for the multi-depot heterogeneous vehicle routing problem with time windows". In: *Transport Research Part C: Emerging Technologies* 19.5, pp. 723–740.

Caceres, J., P. Arias, D. Guimarans, D. Riera, and A. A. Juan (2014). "Rich vehicle routing problem: a survey". In: *ACM Computing Surveys* 47.2. ISSN: 0360-0300.

Cattaruzza, D., N. Absi, D. Feillet, and D. Vigo (2014). "An iterated local search for the multi-commodity multi-trip vehicle routing problem with time windows". In: *Computers & Operations Research* 51, pp. 257–267.

Ceselli, A., G. Righini, and M. Salani (2009). "A column generation algorithm for a rich vehicle routing problem". In: *Transport Science* 43, pp. 56–69.

Chan, Y., W.B. Carter, and M.D. Burnes (2001). "A hybrid algorithm for multi-depot vehicle routing problem". In: *Computers & Operations Research* 28.8, pp. 803–826.

Chao, I., B. Golden, and E. Wasil (1993). "A new heuristic for the multi-depot vehicle routing problem that improves upon best known solutions". In: *American Journal of Mathematical and Management Sciences* 13.3-4, pp. 371–406.

Chen, P. and X. Xu (2008). "A hybrid algorithm for multi-depot vehicle routing problem". In: *IEEE International Conference on Service Operations and Logistics, and Informatics*, pp. 2031–2034.

Chen, Y.-K., C.-Y. Wang, and Y.-Y. Feng (2010). "Application of a 3NN+1 based CBR system to segmentation of the note book computers market". In: *Expert Systems with Applications* 37, pp. 276–281.

Clarke, G. and J. Wright (1964). "Scheduling of vehicles from a central depot to a number of delivering points". In: *Operations Research* 12, pp. 568–581.

Contardo, C. and R. Martinelli (2014). "A new exact algorithm for the MDVRP under capacity and route length constraints". In: *Discrete Optimization* 12, pp. 129–146.

Cordeau, J.F. and M. Maischberger (2012). "A parallel iterated tabu search heuristic for vehicle routing problems". In: *Computers & Operations Research* 39.9, pp. 2033–2050.

Cordeau, J.F., M. Gendreau, and G. Laporte (1997). "A tabu search heuristic for periodic and multidepot vehicle routing problems". In: *Networks* 30.2, pp. 105–119.

Datta, Y. (1996). "Market segmentation: an integrated framework". In: *Long Range Planning* 29, pp. 797–811.

DeSarbo, W.S. and C.F. DeSarbo (2007). "A generalized normative segmentation methodology employing conjoint analysis". In: *Conjoint Measurement: Methods and Application*. Ed. by A. Gustafsson, A. Herrmann, and F. Huber. Springer Science & Business Media, pp. 321–345.

Ehmke, J.F., A.M. Campbell, and T.L. Urban (2015). "Ensuring service levels in routing problems with time windows and stochastic travel times". In: *European Journal of Operational Research* 240.1, pp. 539–550.

Fish, K.E., J.H. Barnes, and M.W. Aiken (1995). "Artificial neural networks: a new methodology for industrial market segmentation". In: *Industrial Marketing Management* 24.5, pp. 431–438.

Foedermayr, E.K. and A. Diamantopoulos (2008). "Market segmentation in practice: review of empirical studies, methodological assessment, and agenda for future research". In: *Journal of Strategic Marketing* 16.3, pp. 223–265.

Geetha, S., P.T. Vanathi, and G. Poonthalir (2012). "Metaheuristic approach for the multi-depot vehicle routing problem". In: *Applied Artificial Intelligence* 26.9, pp. 878–901.

Gillett, B.E. and J.G. Johnson (1976). "Multi-terminal vehicle-dispatch algorithm". In: *Omega* 4.6, pp. 711–718.

Golden, B.L., T.L. Magnanti, and H.Q. Nguyen (1977). "Implementing vehicle routing algorithms". In: *Networks* 7.2, pp. 113–148.

Hastie, T., R. Tibshirani, and J. Friedman (2009). *The elements of statistical learning*. 2nd ed. Springer.

Ho, W., G.T.S. Ho, P. Ji, and H.C.W. Lau (2008). "A hybrid genetic algorithm for the multi-depot vehicle routing problem". In: *Engineering Applications of Artificial Intelligence* 21.4, pp. 548–557.

Huang, J.-J., G.-H. Tzeng, and C.-S. Ong (2007). "Marketing segmentation using support vector clustering". In: *Expert Systems with Applications* 32.2, pp. 313–317.

Juan, A. A., J. Faulin, R. Ruiz, B. Barrios, M. Gilibert, and X. Vilajosana (2009c). "Using oriented random search to provide a set of alternative solutions to the capacitated vehicle routing problem". In: *Operations Research and Cyber-Infrastructure*. Springer, New York, USA, pp. 331–346.

Juan, A. A., J. Faulin, R. Ruiz, B. Barrios, and S. Caballe (2010). "The SR-GCWS hybrid algorithm for solving the capacitated vehicle routing problem". In: *Applied Soft Computing* 10.1, pp. 215–224.

Juan, A. A., I. Pascual, D. Guimarans, and B. Barrios (2015c). "Combining biased randomization with iterated local search for solving the multidepot vehicle routing problem". In: *International Transactions in Operational Research* 22.4, pp. 647–667.

Kotler, P. and G. Armstrong (2011). *Principles of marketing*. 14th. Pearson Prentice Hall.

Kuhn, M. (2008). "Building predictive models in R using the caret package". In: *Journal of Statistical Software* 28.5.

Kuo, R.J., L.M. Ho, and C.M. Hu (2014). "Integration of self-organizing feature map and K-means algorithm for market segmentation". In: *Computers & Operations Research* 29.11, pp. 1475–1493.

Lahrichi, N., T.G. Crainic, M. Gendreau, W. Rei, G.C. Crişan, and T. Vidal (2012). *An integrative cooperative search framework for multi-decision-attribute combinatorial optimization*. Tech. rep. Technical report No. 42. CIRRELT.

Lantz, B. (2013). *Machine learning with R*. Packt Publishing.

Li, J., P.M. Pardalos, H. Sun, J. Pei, and Y. Zhang (2015). "Iterated local search embedded adaptive neighborhood selection approach for the multi-depot vehicle routing problem with simultaneous deliveries and pickups". In: *Expert Systems with Applications* 42.7, pp. 3551–3561.

Liu, Y., M. Kiang, and M. Brusco (2012b). "A unified framework for market segmentation and its applications". In: *Expert Systems with Applications* 39.6, pp. 10292–10302.

Lourenço, H.R., O.C. Martin, and T. Stützle (2010). "Iterated local search: framework and applications". In: *Handbook of Metaheuristics*. Ed. by M. Gendreau and J.-Y. Potvin. Vol. 146. International Series in Operations Research & Management Science. Springer US, pp. 363–397.

Luo, J. and M.-R. Chen (2014a). "Improved shuffled frog leaping algorithm and its multi-phase model for multi-depot vehicle routing problem". In: *Expert Systems with Applications* 41.5, pp. 2535–2545.

— (2014b). "Multi-phase modified shuffled frog leaping algorithm with extremal optimization for the MDVRP and the MDVRPTW". In: *Computers & Industrial Engineering* 72, pp. 84–97.

McCarty, J.A. and M. Hastak (2007). "Segmentation approaches in data-mining: A comparison of RFM, CHAID, and logistic regression". In: *Journal of Business Research* 60.6, pp. 656–662.

Mirabi, M., S.M.T. Fatemi-Ghomi, and F. Jolai (2010). "Efficient stochastic hybrid heuristics for the multi-depot vehicle routing problem". In: *Robotics and Computer-Integrated Manufacturing* 26.6, pp. 564–569.

Montgomery, D.C., E.A. Peck, and G.G. Vining (2012). *Introduction to Linear Regression Analysis*. 5th. Wiley Publishing.

Montoya-Torres, J., J. Lopez, S. Nieto, H. Felizzola, and N. Herazo-Padilla (2015). "A literature review on the vehicle routing problem with multiple depots". In: *Computers & Industrial Engineering* 79, pp. 115–129.

Nagy, G. and S. Salhi (2005). "Heuristic algorithms for single and multiple depot vehicle routing problems with pickups and deliveries". In: *European Journal of Operational Research* 162, pp. 126–141.

Ombuki-Berman, B. and F.T. Hanshar (2009). "Using genetic algorithms for multi-depot vehicle routing". In: *Bio-inspired algorithms for the vehicle routing problem*. Ed. by F.B. Pereira and J. Tavares. Springer Berlin Heidelberg, pp. 77–99.

Pisinger, D. and S. Ropke (2007). "A general heuristic for vehicle routing problems". In: *Computers and Operations Research* 34.8, pp. 2403–2435.

Polacek, M., R.F. Hartl, and K. Doerner (2004). "A variable neighborhood search for the MDVRP with time windows". In: *Journal of Heuristics* 10, pp. 613–627.

Raft, O.M. (1982). "A modular algorithm for an extended vehicle scheduling problem". In: *European Journal of Operational Research* 11, pp. 67–76.

Rahimi-Vahed, A., T.G. Crainic, M. Gendreau, and W. Rei (2013). "A path relinking algorithm for a multi-depot periodic vehicle routing problem". In: *Journal of Heuristics* 19.3, pp. 497–524.

Renaud, J., G. Laporte, and F.F. Boctor (1996). "A tabu search heuristic for the multi-depot vehicle routing problem". In: *Computers & Operations Research* 23.3, pp. 229–235.

Salhi, S. and M. Sari (1997). "A multi-level composite heuristic for the multi-depot vehicle fleet mix problem". In: *European Journal of Operational Research* 103, pp. 95–112.

Salhi, S., A. Imran, and N.A. Wassan (2014). "The multi-depot vehicle routing problem with heterogeneous vehicle fleet: Formulation and a variable neighborhood search implementation". In: *Computers & Operations Research* 52, pp. 315–325.

Smith, W. R. (1956). "Product differentiation and market segmentation as alternative marketing strategies". In: *Journal of Marketing* 21, pp. 3–8.

Taş, D., O. Jabali, and T.V. Woensel (2014). "A vehicle routing problem with flexible time windows". In: *Computers & Operations Research* 52, pp. 39–54.

Team, R Development Core (2008). *R: a language and environment for statistical computing*. R Foundation for Statistical Computing.

Thangiah, S. R. and S. Salhi (2001). "Genetic clustering: an adaptive heuristic for the multi-depot vehicle routing problem". In: *Applied Artificial Intelligence* 15.4, pp. 361–383.

Tillman, F. A. (1969). "The multiple terminal delivery problem with probabilistic demands". In: *Transportation Science* 3, pp. 192–204.

Tillman, F. A. and T. M. Cain (1972). "An upper bound algorithm for the single and multiple terminal delivery problem". In: *Management Science* 18.11, pp. 664–682.

Vellido, A., P. J. G. Lisboa, and K. Meehan (1999). "Segmentation of the on-line shopping market using neural networks". In: *Expert Systems with Applications* 17.4, pp. 303–314.

Vidal, T., T. G. Crainic, M. Gendreau, and C. Prins (2013). "A hybrid genetic algorithm with adaptive diversity management for a large class of vehicle routing problems with time windows". In: *Computers & Operations Research* 40.1, pp. 475–489.

Wang, Y. and I.H. Witten (1996). *Induction of model trees for predicting continuous classes*. Tech. rep. Working paper 96/23. Hamilton, New Zealand: University of Waikato.

Wind, Y. (1978). "Issues and advances in segmentation research". In: *Journal of Marketing Research* 15.3, pp. 317–337.

Wu, T.-H., C. Low, and J.-W. Bai (2002). "Heuristic solutions to multi-depot location-routing problems". In: *Computers & Operations Research* 29.10, pp. 1393–1415.

Yu, B., Z.-Z. Yang, and J.-X.Xie (2011). "A parallel improved ant colony optimization for multi-depot vehicle routing problem". In: *Journal of the Operational Research Society* 62, pp. 183–188.

A.3 A statistical learning based approach for parameter fine-tuning of metaheuristics

Laura Calvet[1], Angel A. Juan[1], Carles Serrat[2], Jana Ries[3]

1. Department of Computer Science, Open University of Catalonia, IN3, 08018 Barcelona, Spain
e-mail: {lcalvetl, ajuanp}@uoc.edu
2. Department of Mathematics, Technical University of Catalonia, 08028 Barcelona, Spain
e-mail: carles.serrat@upc.eduoc.edu
3. Portsmouth Business School, University of Portsmouth, PO1 3DE, UK
e-mail: jana.ries@port.ac.uk

Abstract

Metaheuristics are approximation methods used to solve combinatorial optimization problems. Their performance usually depends on a set of parameters that need to be adjusted. The selection of appropriate parameter values causes a loss of efficiency, as it requires time, and advanced analytical and problem-specific skills. This paper provides an overview of the principal approaches to tackle the Parameter Setting Problem, focusing on the statistical procedures employed so far by the scientific community. In addition, a novel methodology is proposed, which is tested using an already existing algorithm for solving the Multi-Depot Vehicle Routing Problem.

Keywords: Parameter Fine-Tuning, Metaheuristics, Statistical Learning, Biased Randomization.

1. Introduction

Mathematical optimization plays an important role both in research and in our everyday lives. Management of portfolios, vehicle routing or DNA sequence assembly, are only some of the fields in which optimization techniques are employed.

Most of the existing proposals to solve optimization problems can be classified into exact methods or heuristic/metaheuristic approaches (Talbi, 2009). The former guarantee the optimality of the solution found. Unfortunately, a number of relevant problems are particularly complex, and tackling them with state-of-the-art exact methods would require substantial computer memory and time. Problems of this kind are known to be NP-hard (Bovet and Crescenzi, 1994). The Facility Location Problem, the Knapsack Problem and the Multi-Depot Vehicle Routing Problem (MDVRP) are some examples of NP-hard problems. In these cases, heuristics present some experience-based techniques that implement strategies to obtain a sufficiently good solution in a reasonable amount of time. Although they do not provide any theoretical guarantee, they are a popular choice when solving NP-hard problems. Owing to its nature, any heuristic is problem-dependent, which restricts its application to one particular class of problems. Also, heuristics usually provide sub-optimal solutions. These factors have led to the introduction of metaheuristics.

Birattari and Kacprzyk (2009) defines metaheuristics as "general algorithmic templates that can be easily adapted to solve the most different optimization problems". A number of them are nature-inspired, include stochastic components and have several parameters (Boussaïd et al., 2013). They are present in a large number of research areas as telecommunications (Martins and Ribeiro, 2006), machine learning (Carvalho et al., 2011), and vehicle routing (Gendreau et al., 2008), among others.

Although the performance of metaheuristics is known to depend on its parameter values, the scientific community has not formally addressed the so-called Parameter Setting Problem (PSP) until the end of the last century. According to Eiben et al. (1999), during the first decades of metaheuristics research, many scientists based their choices on tuning the parameters "by hand", i.e., experimenting with different values and selecting the ones that provide the best outputs, or "by analogy", applying settings that have been proven successful for similar problems. More recently, the need for a systematic approach towards setting of metaheuristic parameters has been increasingly outlined in the literature (Hooker, 1995; Johnson, 2002). Subsequently, researchers

employ a scientific approach to tackle the PSP more frequently. It is important to highlight that the selection of a systematic methodology leads to a gain of efficiency, as in general, less time is required to fine-tune the parameters while the performance of the metaheuristic is the same if not improved. However, there is no methodology commonly accepted by the scientific community and there is also a lack of publications that compare, in an exhaustive and objective manner, the main approaches and the techniques used so far. Moreover, some of the proposed methodologies are not easily reproducible or are highly metaheuristic and problem dependent. These are some of the reasons why, in spite of the amount of parameter fine-tuning works, many practitioners go on tuning by hand or designing algorithms without parameters (or with a very low number of them), even in the case when more parameterized algorithms could lead to better performances.

This article aims to contribute to the literature by proposing a general and automated statistical learning based procedure to tackle the PSP. It is accompanied by some methodological guidelines to validate the results. In order to test the methodology and illustrate its application, the approach is employed to fine-tune a hybrid algorithm implemented to solve the MDVRP. The remainder of this article is organized as follows. Section 2 presents a formal definition of the PSP, the existing approaches, and their main contributions. Our methodology is outlined in Section 3, followed by Section 4, which shows its application on a hybrid algorithm. A discussion of the results is reported in Section 5. Finally, Section 6 presents concluding remarks.

2. Related work on the Parameter Setting Problem

Ries et al. (2012) define the PSP as the search for a set of parameter values θ^* in the parameter space Θ such that $\forall\theta \in \Theta : \theta^* \succeq \theta$ (where $\succeq$ denotes a relation of preference), for a given metaheuristic m in the metaheuristic space M, and a given instance x or group of them X in the instance space I. In practice, the amount of time available for experimenting T may be a restriction. In this case, the solution is approximate ($\hat{\theta}$). With regards to the difficulty of this problem, Montero et al. (2014) states that: (*a*) it is time consuming; (*b*) the best set of parameter values depends on the problem at hand; and (*c*) the parameters can be interrelated.

During the last decades, a large number of methodologies have been put forward to solve the PSP. These proposals can be classified in two groups (Birattari and Kacprzyk, 2009): Parameter Control Strategies (PCS), and Parameter Tuning Strategies (PTS). This classification is extended by Instance-specific Parameter Tuning Strategies (IPTS), which includes features of the aforementioned groups.

This section provides a brief description of each approach and some of the most cited works. We refer the interested reader to more specific publications such as Eiben et al. (1999), De Jong (2007) and Battiti and Brunato (2010) for an expanded review of PCS, Birattari and Kacprzyk (2009) in the case of PTS, and Ries (2009) for IPTS.

Parameter Control Strategies (PCS)

These methodologies aim for a dynamic fine-tuning of the parameters by controlling and adapting their values while solving a problem instance. They follow two basic steps: firstly, an initial set of parameter values is chosen; secondly, an adaptation mechanism is integrated which changes relevant parameter values. Most of these strategies apply Adaptive Parameter Control, which means that their adaptation mechanism is based on the assessment of particular information that is stored during the iterative process of a metaheuristic. This information is usually related to the goodness of intermediate solutions. Figure A.1 outlines the main instructions of a PCS based on Adaptive Parameter Control. The main drawbacks of this approach are the potentially high computational effort required and the lack of acquired understanding about good parameter values each time an instance is solved.

Eiben et al. (1999) addressed the PSP in Evolutionary Algorithms (EAs). Three categories were defined to classify the PCS. The first one, Deterministic Parameter Control, alters the value of a parameter by some deterministic rule, which is usually time based. The second category, Adaptive Parameter Control, does employ feedback to determine the direction and/or magnitude of a parameter change. This is the most used kind of control. Consequently, we will focus on it. The third, Self-Adaptive Parameter Control (Smith, 2008), encodes the parameters to be adapted into the chromosomes of an EA. De Jong (2007) described the main motivations to use dynamic parameter setting strategies in EAs: first, as the running proceeds, information about the fitness

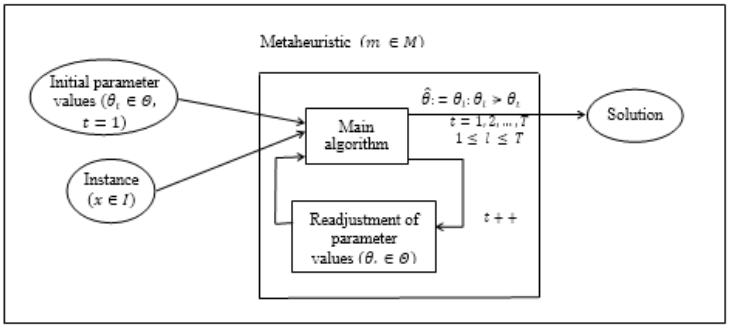


FIGURE A.1: Scheme of PCS applying an Adaptive Parameter Control.

landscape is generated, which may be used to improve the performance; also, changing the parameters is needed as an EA "evolves from a more diffuse global search process to a more focused converging local search process".

Table A.1 gathers a few representative works following this approach. Nowadays, it constitutes a popular choice, mostly in EAs. From the literature, it can be concluded that the parameter fine-tuning is a difficult task, partly due to the potential interactions between parameters (Eiben et al., 1999; De Jong, 2007; Smith, 2008). The worth of applying PCS is sometimes doubted (Beasley et al., 1993) or not recommended for static optimization problems (De Jong, 2007). However, most authors agree that this approach has a long way to go.

TABLE A.1: Representative works employing PCS.

Work	Main techniques	Metaheuristic	Optimization problem
Battiti and Tecchiolli (1994) and Battiti and Brunato (2005)	Reactive Scheme	Tabu Search (TS)	Quadratic Assignment Problem (QAP), and Maximum Clique Problem
Zennaki and Ech-Cherif (2010)	Support Vector Machines	TS	TSP
Lessmann et al. (2011)	Regression Models	Particle Swarm Optimization (PSO)	Water Supply Network Planning Problem

Parameter Tuning Strategies (PTS)

This approach relies on the concept of robustness (Viana et al., 2005). A robust algorithm provides good results for a given set of instances of a problem using a fixed set of parameter values. The basic procedure (Figure A.2) involves finding a set of parameter values providing satisfactory results for a set of instances, usually using statistical and/or optimization techniques. Some authors analyse only a representative subset of instances and apply the set of parameter values found to solve all the instances. This approach also includes the case of solving one instance.

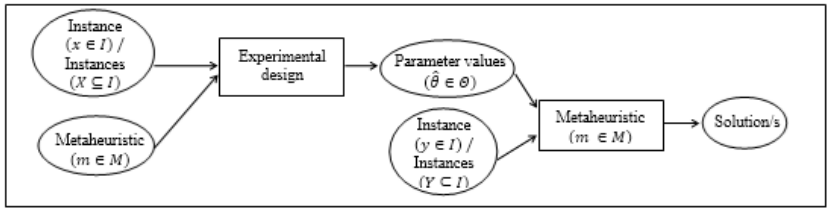


FIGURE A.2: Scheme of PTS.

The work of Czarn et al. (2004) is an outstanding contribution from a statistical point of view. It addresses the issues of blocking when using Design of Experiments (DOE) for variation or noise due to seed, testing individual parameters and interactions, and performing power analyses, among others.

Table A.2 shows some works relying on this approach. Many authors focus on minimizing the number of runs, presenting simple models without interactions (e.g., Coy et al., 2001; Pongcharoen et al., 2007; Xu et al., 1998). DOE and regression analysis are the most employed techniques. The main criticism these works may receive is that most need an initialization of methodology-specific parameters that in some cases is not fully reported. Fortunately, the number of papers that report applications of their methodology in more than one problem or in real-world problems is increasing.

TABLE A.2: Representative works implementing PTS.

Work	Main techniques	Metaheuristic	Optimization problem
Park and Kim (1998)	Simplex method	SA	Graph Partitioning Problem, Permutation Flow Shop Scheduling Problem, and Short-term Production Scheduling Problem
Xu et al. (1998)	Tree growing and pruning method based on statistical tests	TS	Steiner Tree-Star Problem
Coy et al. (2001)	DOE and Linear Regression	Routing heuristics	Vehicle Routing Problem
Bartz-Beielstein et al. (2004)	DOE, Classification and Regression Trees, and Design and Analysis of Computer Experiments	PSO and Nelder-Mead Simplex Algorithm	Elevator Group Controller Problem
Ramos et al. (2005)	Logistic Regression	EA	TSP
Birattari and Kacprzyk (2009) and Birattari et al. (2010)	Racing Algorithm (Maron and Moore, 1993) and the Friedman's two-way analysis of variance by ranks (Conover, 1999)	Iterated Local Search (ILS) and Ant Colony Optimization (ACO)	QAP and TSP
Adenso-Diaz and Laguna (2006)	DOE and Local Search	Neighbourhood structure, TS, SA, TS, Heuristic based on the SA and the TS, and TS	Steiner Problem, Part-Machine Grouping Problem, Part-Machine Grouping Problem, Single-Machine Scheduling, Proportionate Flowshops, and Bandwidth Packing
Pongcharoen et al. (2007)	DOE	GA	TSP
Ridge and Kudenko (2007)	DOE and Desirability Functions	ACO	TSP
Gunawan et al. (2013)	DOE, Response Surface Methodology and ParamILS (Hutter et al., 2009)	SA	Industry Spares Inventory Optimization Problem

Instance-specific Parameter Tuning Strategies (IPTS)

As in the case of PCS, IPTS aim for an instance-specific tailoring of the parameters. At the same time, these strategies use a fixed set of parameter values, as the PTS, avoiding the need of modifying the metaheuristic algorithm and reducing the potential computational effort required to adapt parameter values during the algorithmic run. In order to implement these strategies the relation between the parameter values and the performance of the metaheuristic has to be analysed, taking into account instance features. The next step consists in developing a mechanism able to use the features of a new instance to recommend a set of parameter values. The key element is the selection of instance features easy and fast to compute, and good at discriminating instances on the shape of their fitness landscapes, which analyse the relationship between the objective function values and the parameters. This learning may take a non-negligible amount of time, but it is assumed that this approach requires less computational time than the PCS approach does. The procedure is shown in Figure 3.

Some contributions are included in Table A.3. The number of works is low since it is relatively new. As in the previous cases, they employ a variety of techniques and analyse several problems.

It has been seen that the literature on the PSP is relatively diverse. However, more research is needed to fully explore and compare the performance of different techniques from statistics and

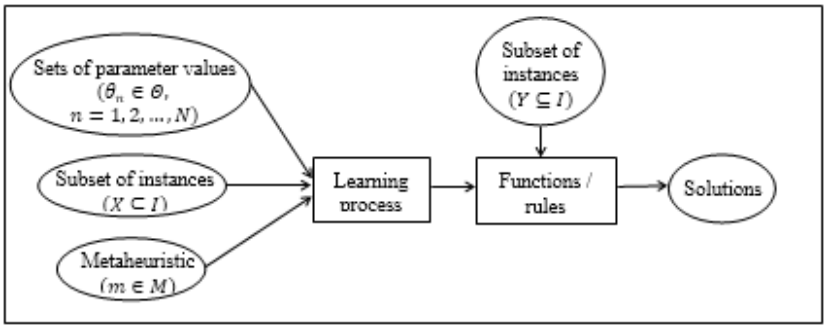


FIGURE A.3: Scheme of IPTS.

TABLE A.3: Representative works implementing PTS.

Work	Main techniques	Metaheuristic	Optimization problem
Ries (2009)	DOE and Fuzzy Logic	Guided Local Search and GA	TSP
Pavón et al. (2009)	Case-Based Reasoning and Bayesian Networks	GA	Root Identification Problem
Dobslaw (2010)	DOE and Artificial Neural Networks	PSO	TSP

operations research (OR), and to achieve that researchers and practitioners become aware of the relevant effect that an adequate parameter-fine tuning may have. In this paper we mainly focus on the parameter fine-tuning of metaheuristic algorithms from an OR perspective. Notice, however, that the literature on parameter fine-tuning of general algorithms is much more extensive, and it has been mainly developed by the computer science community. This community addresses a larger variety of problems (not only of optimization nature), tends to employ algorithms with a larger number of parameters, and uses to consider more complex and/or time-consuming approaches for setting the parameters of different types of algorithms -including searching and classification algorithms, etc. Thus, for example, Ansótegui et al. (2015) or Hutter et al. (2011) describe general but complex methods that can be used in the fine-tuning process of several types of algorithms. These general approaches are rarely considered by the OR community. Accordingly, one of the main contributions of this paper is to provide the OR community with an alternative methodology, which is easier to use and faster, and that can be employed to simplify and make more agile the fine-tuning process of metaheuristic algorithms.

Approaches comparison

All approaches have different advantages. The dynamic adaptation of the parameter values that characterizes PCS usually provides better results. However, the computational effort tends to be higher. On the other hand, the PTS approach is the easiest and fastest to use, once a set of parameter values is selected. Although the code of the algorithm is not changed, finding an adequate set may be also time-consuming. The last group of strategies represents a compromise solution: it takes less computational time than the PCS approach, but requires implementing a learning mechanism, for which statistical learning skills are needed. Therefore, there is no approach that stands out from the others. Probably, the most adequate depends on the specific problem to tackle, the instances to solve, the available time and the skills of the researcher. Despite this fact, some general guidelines can be formulated. PTS can be considered as the best option when working with robust algorithms. Regarding IPTS, they are more complex than PTS but provide better results when the algorithm is not robust. In case of prioritizing the algorithm performance, PCS usually constitute the most recommendable approach.

3. Our approach

We propose a methodology that follows the PTS approach. There are several reasons for choosing it. Firstly, it is not computationally intensive, since it may focus on a subset of instances. The

inference from a representative sample of benchmark instances to the whole set usually provides good results, specifically if the analysed algorithm is robust. There are two conditions that imply robustness. First, the algorithm has to be little sensitive to small changes in the parameter values, and second, the fitness landscapes for different instances have to be similar. These conditions guarantee that the best set of parameter values for one instance will probably provide good results for the others. The high number of works following this approach, which cover several metaheuristics and optimization problems, shows that many metaheuristic algorithms can be considered robust. Another reason for focusing on PTS is that there is no methodology based on this approach and widely employed, but at the same time, there are plenty of techniques that can be used. Some of them have been intensively tested as DOE and Regression Analysis. However, others remain to be investigated.

Our methodology is based on clustering (Hastie et al., 2009) and DOE (Montgomery, 2008). These are two well-established techniques that can be easily implemented using free statistical software. The clustering groups instances that have a similar fitness landscape. It facilitates the selection of representative instances and also provides information that can be used to perform a more flexible fine-tuning if each group is treated independently, i.e., exploring the fitness landscape of an instance to find a good set of parameter values and applying it to solve the instances assigned to the same group. Regarding DOE, it enables experimenters to identify and quantify the effects of several parameters and their interactions on the objective function value.

The remainder of this section presents a statistical learning based methodology to obtain a list of sets of parameter values, and a more global procedure to validate and assess its goodness.

General methodology

A 4-step procedure is exposed herein. It is assumed that the experimenter has described and modelled a problem, and has chosen the metaheuristic to tackle it and a set of benchmark instances.

- The first step involves choosing a subset of the instances. Their fitness landscapes will be analysed in order to obtain sets of parameter values that provide good results for them. The subset has to be representative as these sets of parameter values will be used to solve the whole set of instances. An approach to select a representative subset is, firstly, to determine the instance features that have a major influence on which set of parameter values is the most adequate, and then, choose the instances in such a way that the feature values of the subset are representative of those of the entire set of instances. For example, if we have a parameter for which its optimum value is known to depend on the instance size, a representative subset of the instances will present the same proportion of instances of a given size that the whole set does. This approach can be particularly difficult when there are several non-independent parameters. A possible simplification for feature selection consists of choosing those that are commonly used to discriminate instances of a specific problem. Several examples can be found in the literature. Coy et al. (2001) considered, when addressing the Capacitated Vehicle Routing Problem (CVRP), the distribution of customers, the distribution of demand and the location of the depot. Ries et al. (2012) studied the size, the distance metric, a ratio to describe the shape of the area within which a set of cities is distributed and a measure of clustering for the TSP.

  In contrast, a problem-independent approach is proposed here. Initially, for a given number of randomly generated sets of parameter values, each instance is solved several times using different seeds for the random number generator of the algorithm (or only once if the algorithm is deterministic). Alternatively, the sets could also be generated using more advanced statistical techniques such as DOE. We consider the median of the objective function values found with the same parameter values but different seeds. The median is a robust measure to aggregate data, but many others could be employed. It is essential to remark the importance that a seed may have in the performance of an algorithm (Juan et al., 2015c; Czarn et al., 2004). Afterwards, feature scaling is applied to the values obtained for each instance. Then, this data is used to cluster instances and select a representative one from each cluster. These instances form the subset to analyse.

  Although it is a computationally intensive approach, we think it is effective to assess which instances show a similar relation between parameter values and the performance of an algorithm.

For each instance of the subset, the steps ranging from the second to the fourth are implemented as follows.

- The second step requires selecting the range over which each parameter can be set. Some experience or knowledge about the problem and the metaheuristic may be highly valuable. The ranges should be large enough to cover at least one set of parameter values that can provide a sufficiently good solution with a high probability. On the other hand, a smaller range would allow the experimenter to describe more accurately, with the same resources, the relationship between the parameter values and the objective function value. If there is no a priori information about which are the best regions of the parameter space, a suitable procedure is to perform a rough and fast landscape analysis. Specifically, some possible combinations of parameter values can be selected and utilised to run the algorithm. The best results will identify promising regions. There are several ways of choosing the combinations, as equally-spaced or randomly generated sets. This analysis holds a trade-off between the computational time required and the reliability of the conclusions.

- The third step consists of designing an experiment. A Central Composite Design is studied. Each metaheuristic parameter is considered a factor and the extreme values of its range define the levels. According to this design, the algorithm is executed also several times for each combination of factor values, each one with a different seed.

- In the fourth step, a procedure is developed to search the neighbourhood of the best set of parameter values found. Specifically, another Central Composite Design centred on this set is applied.

Finally, the upshot is a list of recommended sets of parameter values, one per cluster; in particular, those that reported the best results on the last step. The procedure is shown in Figure A.4.

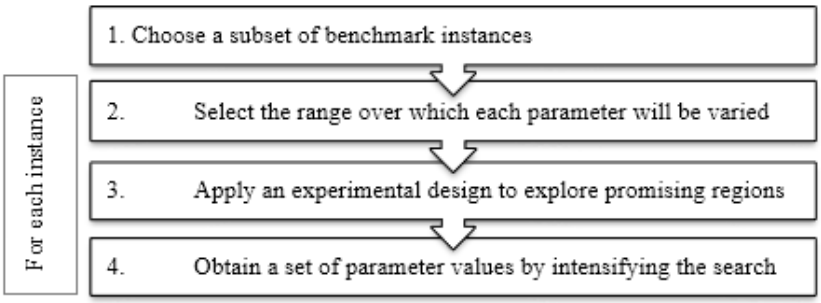


FIGURE A.4: Outline of the procedure for parameter fine-tuning.

An extended proceeding (Figure A.5) is described below in order to validate the list of sets of parameter values obtained and analyse the results provided by it.

Before all else, a list of sets of parameter values, $\hat{\theta} = (\hat{\theta}_1, \hat{\theta}_2, \ldots, \hat{\theta}_K)$ where K is the number of clusters, is chosen as has been explained in the precedent section. Later on, each instance of the subset used to select $\hat{\theta}$ is solved with the corresponding set of $\hat{\theta}$, and with different sets, $\bar{\theta}_j$ ($j = 1, 2, \ldots, J$) (equally spaced, randomly selected or relatively close to the set of $\hat{\theta}$ according to some distance measure). To assess the performance of a set of $\hat{\theta}$ in a specific instance regarding the other sets, the associated solutions are compared. Given a decision level parameter r ($1 \leq r \leq J + 1$), if the rank of the objective function value provided by the proposed set is equal or lower than r, then it is considered a good set for that instance. Once all the instances of the subset are examined, it can be reckoned the proportion of them in which the corresponding set has been classified as good. $\hat{\theta}$ is validated by comparing this proportion with a predefined parameter p ($0 < p < 1$); if the proportion is higher, then the experimenter has enough evidence of the quality of $\hat{\theta}$ to go on to test it with other instances in the next step.

If $\hat{\theta}$ is not validated, the process has to be readjusted and restarted. This readjustment may be done in several ways, some options are: checking the robustness and the adequacy of the clustering, adapting the ranges, dedicating more resources to the search, etc. The best strategy is problem-dependent. As a consequence, the choice should rely on the opinion of the experimenter, who will have acquired valuable information from the outputs observed.

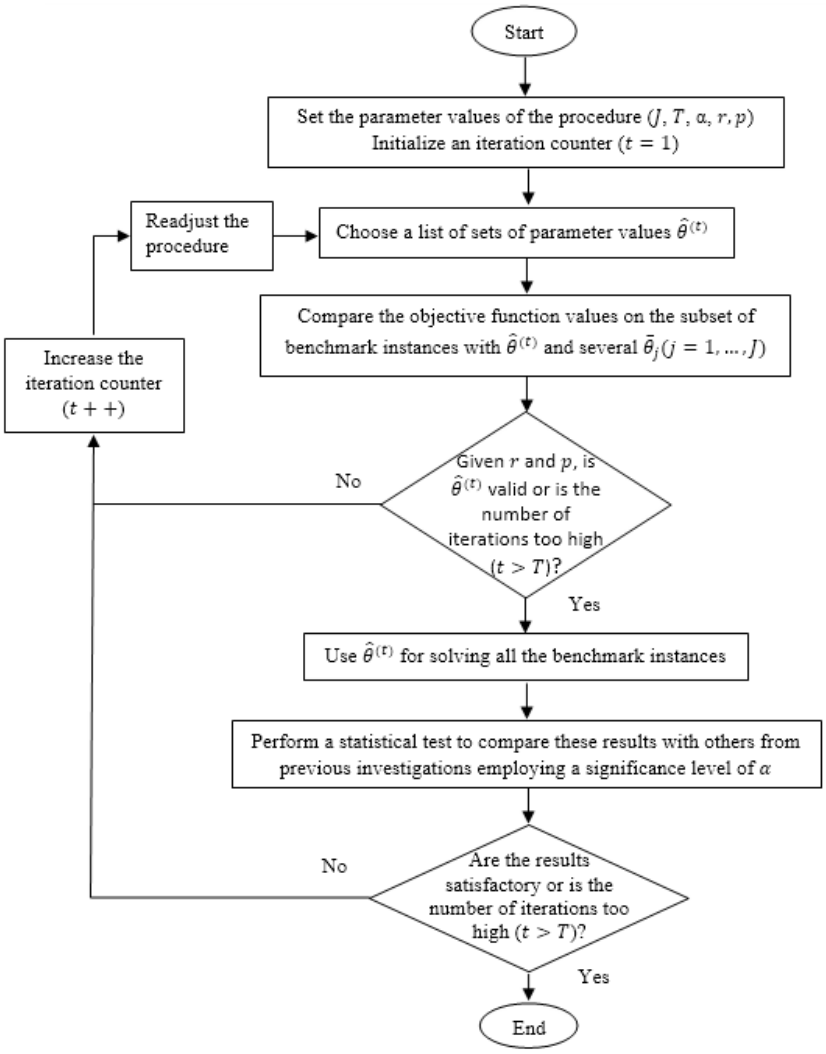


FIGURE A.5: Flowchart representing the proposed methodology.

Once the list of sets of parameter values has been labelled as valid, it is applied for solving the other instances (each one with the set proposed for the representative instance of the cluster where it has been assigned). To examine the effectiveness of the procedure, it is desirable to compare the solutions with others reported in the literature for the same instances, by performing the t-test for paired samples if data is normal, or the Wilcoxon signed rank test otherwise. If the means (or the mean ranks if data is not normal) do not differ significantly, it may be classified as a satisfactory outcome as it will mean that the proposed methodology, automated and general, has been proven to be competitive. If the results are unsatisfactory, the procedure should be modified and reinitiated.

It is useful to consider that, since the available resources are usually limited, the possible readjustments should be also limited (T represents this limit). Consequently, the process may end without a satisfactory list of sets of parameter values. In this case, the list which provides on average the best solutions will be accepted.

4. Experimental results

Case study: Biased Randomization and ILS for solving the Multi-Depot Vehicle Routing Problem (MDVRP)

In order to test our methodology, it was implemented to fine-tune the parameters of the hybrid algorithm described in Juan et al. (2015c), which combines Biased Randomization and the ILS metaheuristic to address the MDVRP. A brief introduction to both the problem and the algorithm are presented in this subsection.

The MDVRP is a variant of the well-known CVRP that consists in planning routes to service a number of customers with a homogeneous fleet of vehicles that have a maximum capacity. All routes begin and end at one depot, where all resources are initially located. The objective is to find a solution (Figure A.6) that minimizes the total cost while satisfying the associated constraints. Typically, these constraints imply that a single vehicle supplies each customer and it cannot stop twice at the same customer. The MDVRP integrates an allocation problem, in which the customers are assigned to one depot, with several CVRPs, one per depot. In the test case, there is also a maximum number of vehicles per depot and a maximum route length. It is considered a challenging problem as allocation and routing issues are interrelated.

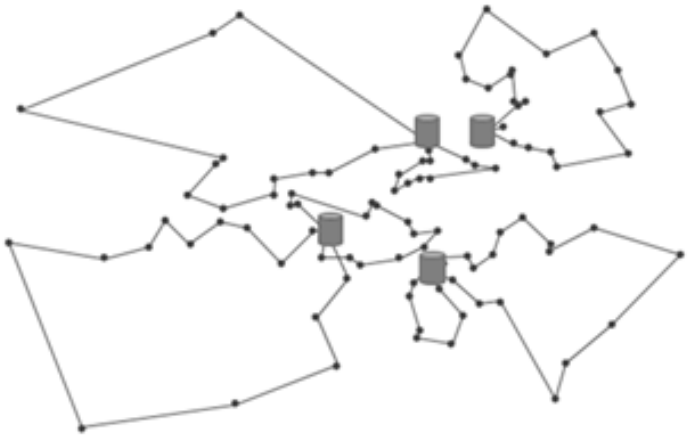


FIGURE A.6: Solution for a medium-size MDVRP with 4 depots (cylinders).

The algorithm follows several steps. Initially, a priority list of potentially eligible customers is computed for each depot. The lists are sorted according to a distance-based criterion. Then, they are randomized based on a geometric distribution and used to allocate customers to depots. Afterwards, an initial solution is built by solving each routing problem independently with a version of the Clarke & Wright's Savings (CWS) heuristic (Clarke and Wright, 1964). In short, CWS starts building an initial solution in which each route includes just one customer. Following that, the heuristic considers the possibility of merging two routes if the total cost is reduced. This operation is repeated until no more merges are possible. For this project, the authors developed a biased-randomized version (Juan et al., 2011a); while the original seeks always the best possible merging, this version applies biased randomization to select one merging (i.e., multiple solutions can be obtained). In the next phase, an ILS procedure is implemented. A new solution is computed by perturbing the current solution, which implies the reallocation of a given percentage of customers. The new solution replaces the current solution if the former is better. If it is also better than the best solution found so far, the latter is updated. On the other hand, if the new solution is worse than the current one, an acceptance criterion is applied and, consequently, the current base solution can still be modified. This phase ends after a fixed number of iterations. Finally, a post-optimization process is applied to the five best solutions.

This algorithm has three main parameters:

- bM: the parameter of the distribution assigning nodes to depots.
- bR: the parameter of the distribution selecting edges in the CWS heuristic.
- $p*$: the percentage of nodes that are reallocated in the ILS phase.

Note that these parameters take values between 0 and 1.

5. Implementation details

The first step is the selection of a representative subset of instances. Initially, 10 randomly generated sets of parameter values, 7 seeds and the 33 benchmark instances solved in Juan et al. (2015c) were selected. Therefore, information from 2310 runs was stored. Data from different seeds was aggregated by computing the median; then feature scaling was applied. The instances that were

considered easy-to-solve, those that presented no variation in the results, were separated. This was done to focus the analysis on the instances for which results could be improved by fine-tuning the parameters. Afterwards, a clustering using the k-medoids algorithm (Theodoridis and Koutroumbas, 2009) was performed. The range of values considered for setting the value of k was 2-12. The final value was selected employing the average silhouette criteria (Rousseeuw, 1987). The composition of the clusters and the representative instances (or medoids) can be observed in Table A.4.

TABLE A.4: Clustering of the benchmark instances.

Medoids	Clusters
p01	p01
p07	p04, p07, p11, p18, pr02, pr05, pr09
p09	p03, p09, pr04, pr10
p17	p17
p19	p19
p22	p22
p23	p20, p23
pr06	p05, p06, p08, p10, p15, pr01, pr03, pr06, pr07, pr08

Once the subset of instances was formed, the second step, setting the ranges of the parameters, was carried out. After a statistical analysis, it was concluded that just two parameters, bM and bR, did significantly affect the performance of the algorithm. Therefore, only those two parameters were studied. Five equally spaced values ranging from 0 to 1 were analysed for each parameter. Each instance was solved seven times (considering different seeds) for each possible combination of parameter values. The objective function values were aggregated as before. Then, the values for other possible combinations were estimated by linear interpolation.

The ranges were set to cover the smallest rectangular area of the parameter space that included the lowest objective function values. In particular, the values labelled as the lowest were those meeting the following condition:

$$\textit{Objective solution} \leq \textit{minimum value} + \beta \cdot (\textit{maximum value} - \textit{minimum value})$$

The value of β was set at a different value for each instance. More precisely, it was the minimum value that encompassed, at least, 5% of the search space. Figure A.7 shows the contour plot and the area in which the search was intensified for each instance.

The next step was applying a design for each instance of the subset. It was performed to better analyse the relation between the metaheuristic performance and the parameter values. A Face-Centred Central Composite (FCC) Design was selected, as in most of the cases the space parameter could not be expanded (since all parameters could only take values between 0 and 1). Figure A.8 displays the scheme for instance p01. The objective function values for the same instance are represented in Figure A.9.

Then, the neighbourhood of each set that provided the best solution for an instance was explored applying another FCC Design, centred on that set and covering half of the area analysed with the previous design. The sets that finally presented the best performance were stored. They are outlined in Table A.5. Random values were assigned to the instances that did not present variations in the results when changing the parameter values.

TABLE A.5: Proposed list of sets of parameter values.

Medoids	Clusters	bM	bR
p01	p01	0.513	0.501
p07	p04, p07, p11, p18, pr02, pr05, pr09	0.001	0.372
p09	p03, p09, pr04, pr10	0.283	0.283
	p17	random	random
p19	p19	0.443	0.378
p22	p22	0.001	0.231
p23	p20, p23	0.449	0.250
pr06	p05, p06, p08, p10, p15, pr01, pr03, pr06, pr07, pr08	0.500	0.231
	p02, p12, p13, p14, p16, p21	random	random

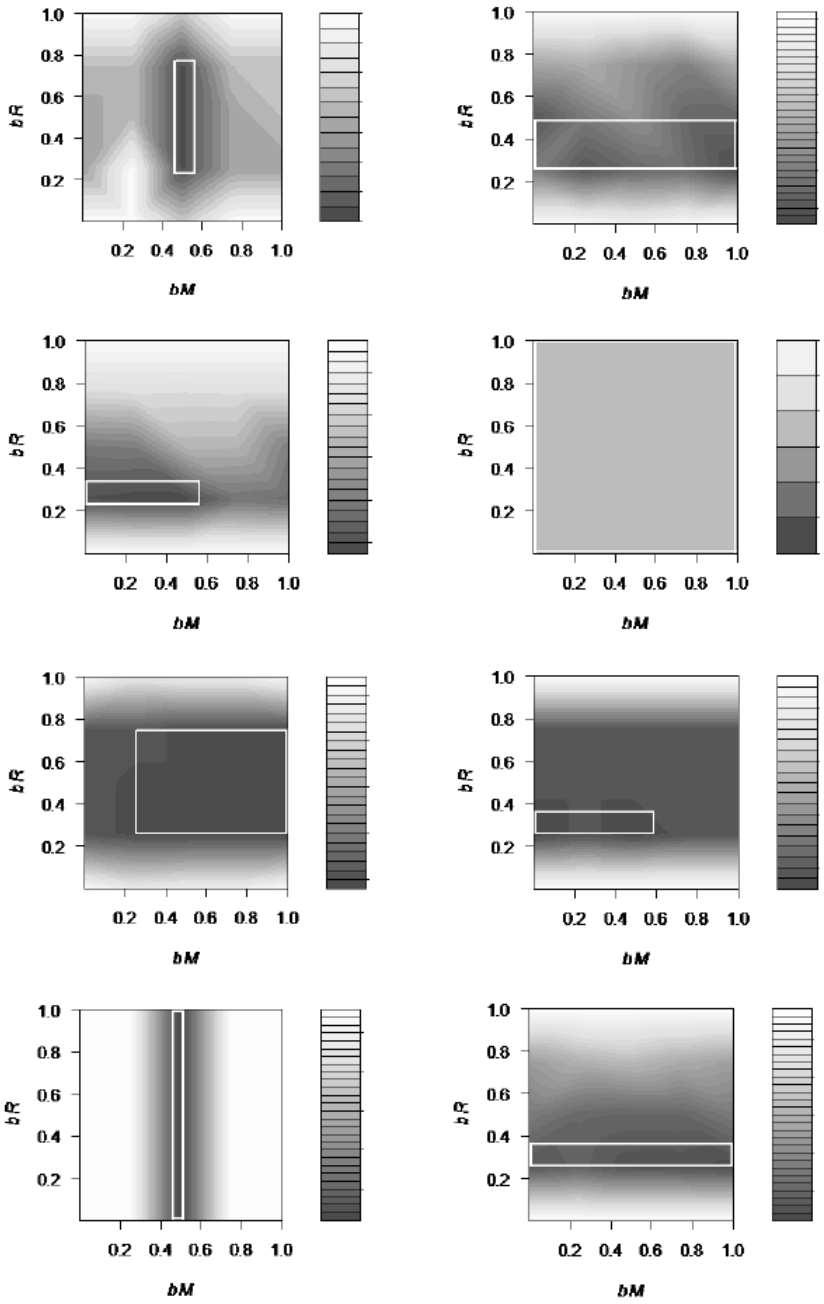


FIGURE A.7: Contour plots of the medoids sorted from left to right, and top to bottom.

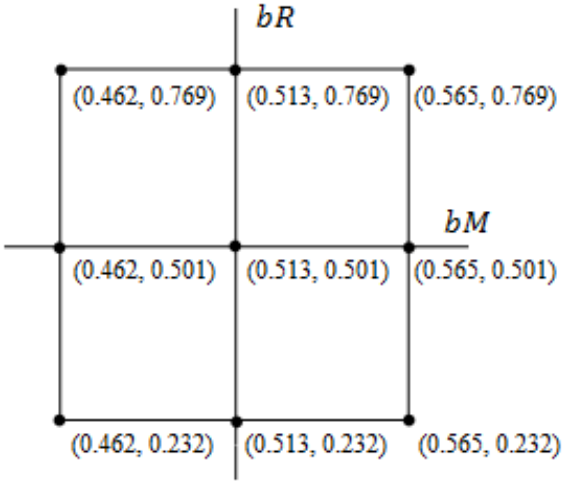


FIGURE A.8: Scheme of the FCC Design applied to the instance p01.

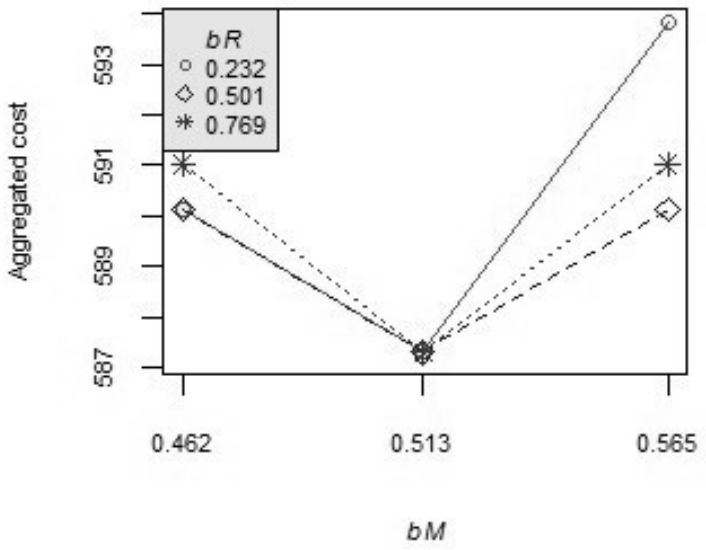


FIGURE A.9: Solutions of the instance p01.

Results

The following parameters were chosen to validate the list of sets: $J = 10$, $T = 3$, $\alpha = 0.05$, $r = 6$, $p = 0.7$. The number of sets randomly generated was fixed considering the trade-off between the reliability of our comparisons and the computational time required. The number of iterations was set considering only the time available. The significance level is the one most commonly used in the literature. The value of the fourth parameter is the mean rank that could be expected due to randomness with 11 solutions (1 set proposed and 10 randomly generated). The last parameter was calibrated to force the algorithm to provide good results at most of the instances. The algorithm was run 7 times with different seeds for each combination of parameter values, the medians and the minimum values were stored. The ranks of the results obtained are detailed in Table A.6. Ties receive a rank equal to the average of the ranks they span, shown inside the parentheses.

Table A.6: Ranks of the results provided by our list and by 10 random sets.

Medoids	Rank (medians)	Rank (minimum values)
p01	1	3.5 (1-6)
p07	5	3.5 (1-6)
p09	2	2
p17	2 (1-3)	1
p19	6.5 (2-11)	10.5 (10-11)
p22	11	11
p23	1.5 (1-2)	1
pr06	5	1.5 (1-2)
Valid instances	0.75	0.75

According to our methodology, the list of sets can be considered valid as it presents a rank equal to or below 6 in 75% of the analysed instances, both considering medians and minimum values. In order to test our results, the algorithm was executed with the parameter values suggested in Juan et al. (2015c). Both series of results are comparable as were obtained using the same computer and stopping criteria based on the number of iterations. Table A.7 presents the parameter values used in the aforementioned paper. Instead of setting fixed values, the authors introduced randomness by employing uniform distributions. The lower and upper bounds were selected after some tests.

Table A.7: Sets of parameter values for comparison.

bM	bR	p*
Uniform (0.5, 0.8)	Uniform (0.1, 0.2)	Uniform (0.1, 0.5)

Table A.8 shows the results obtained solving all instances with the proposed list of sets (our results, OR), and with the set proposed in Juan et al. (2015c) (JR).

6. Discussion of the results

The comparison of the solutions shows that our procedure achieves better results in most of the instances. Table A.9 presents the average and the standard deviation of the differences, and the p-values of the test to compare the mean ranks of the results. It is a non-parametric test as the null hypothesis of the Shapiro-Wilk test, a test of normality, was rejected in all cases. The means are negatives, indicating that our methodology provides better solutions. The p-values reveal that the differences of the mean ranks are not statistically significant. Even though, the magnitude of the mean difference can be considered relevant in the context of the MDVRP.

Results on all instances except the subset of representative instances selected initially and those not analysed because of the null variation of their results allow us to demonstrate the good performance of our methodology, which is not directly attributed to the instances deeply studied but to their representativeness, without considering the changes in the instances that where discarded, which are due to randomness.

TABLE A.8: Instances experimental results.

Inst.	OR medians (1)	OR, minimum values (2)	JR, medians (3)	JR, minimum values (4)	% Gap (1) - (3)	% Gap (2) - (4)
p01	585.000	576.866	593.829	576.866	-1.509	0.000
p02	480.261	476.660	480.261	476.660	0.000	0.000
p03	644.464	641.186	649.229	641.186	-0.739	0.000
p04	1022.085	1019.570	1024.473	1024.062	-0.234	-0.441
p05	760.341	756.281	764.325	754.882	-0.524	0.185
p06	882.827	879.072	880.418	879.763	0.273	-0.079
p07	899.709	897.974	906.395	897.974	-0.743	0.000
p08	4440.534	4434.552	4438.407	4426.747	0.048	0.176
p09	3920.743	3906.561	3923.248	3900.274	-0.064	0.161
p10	3706.763	3667.344	3705.012	3687.054	0.047	-0.537
p11	3598.972	3584.691	3592.891	3585.690	0.169	-0.028
p12	1318.955	1318.955	1318.955	1318.955	0.000	0.000
p13	1318.955	1318.955	1318.955	1318.955	0.000	0.000
p14	1360.115	1360.115	1360.115	1360.115	0.000	0.000
p15	2573.393	2556.846	2573.393	2557.528	0.000	-0.027
p16	2605.565	2585.373	2605.565	2600.099	0.000	-0.570
p17	2720.231	2714.663	2725.799	2725.799	-0.205	-0.410
p18	3831.996	3806.783	3835.388	3806.783	-0.089	0.000
p19	3883.686	3883.686	3883.686	3881.427	0.000	0.058
p20	4080.348	4074.779	4091.482	4091.482	-0.273	-0.410
p21	5706.530	5692.789	5701.902	5692.789	0.081	0.000
p22	5808.738	5806.370	5806.480	5786.288	0.039	0.346
p23	6134.441	6128.873	6145.576	6123.306	-0.182	0.091
pr01	861.319	861.318	861.319	861.318	0.000	0.000
pr02	1330.495	1310.679	1331.543	1314.364	-0.079	-0.281
pr03	1813.634	1813.634	1814.452	1813.634	-0.045	0.000
pr04	2084.843	2077.582	2089.785	2079.832	-0.237	-0.108
pr05	2379.075	2359.947	2379.797	2368.525	-0.030	-0.363
pr06	2709.792	2693.680	2713.593	2696.504	-0.140	-0.105
pr07	1109.235	1109.235	1109.235	1109.235	0.000	0.000
pr08	1680.896	1674.930	1678.872	1674.594	0.120	0.020
pr09	2148.216	2147.192	2153.317	2142.650	-0.237	0.212
pr10	3016.255	3008.129	3028.606	3014.874	-0.409	-0.224

TABLE A.9: Means and standard deviations of the differences and statistical tests.

		Mean of the differences	Standard deviation of the differences	P-value of the comparison of mean ranks
All instances	Medians	-0.149	0.330	0.954
	Minimum values	-0.070	0.219	0.980
All instances except the studied subset and those not analysed	Medians	-0.117	0.247	0.942
	Minimum values	-0.100	0.217	0.942

7. Conclusions

This paper has addressed the Parameter Setting Problem which, due to the relevance of metaheuristics in a number of fields, is increasingly getting more attention.

We have presented an overview of the main approaches: Parameter Control Strategies (PCS), Parameter Tuning Strategies (PTS), and Instance-specific Parameter Tuning Strategies (IPTS). While PCS dynamically adapt the parameter values during the resolution of an instance, PTS let the parameter values fixed and employ them to solve several instances. IPTS represent a compromise solution, the parameter values are not modified during the search but they can be different for each instance, depending on its features. The benefits and pitfalls of each approach have been discussed. In addition, a new methodology which stands out for being automated and, problem- and metaheuristic-independent, has been presented. It incorporates techniques of clustering, which allows splitting the set of instances and, as a consequence, gives more flexibility to the fine-tuning by analysing each subset independently, and Design of Experiments. As a result, we have developed a methodology that avoids the strictness of common PTS, which present only a set of parameter values, and the need of modifying the main algorithm and spending more time on the resolution of instances that characterizes PCS. At the same time, our methodology is simpler than IPTS as it does not require a learning procedure able to recommend an instance-specific set of parameter values. In order to illustrate and test our methodology, it has been applied to a hybrid algorithm. The case study provides promising results.

Acknowledgments

This work has been partially supported by the Spanish Ministry of Economy and Competitiveness (TRA2013-48180-C3-P, MTM2012-38067-C02-01, TRA2015-71883-REDT), and FEDER. Likewise, we want to acknowledge the support received by the Department of Universities, Research and Information Society of the Catalan Government (2014-CTP-00001).

References

Adenso-Diaz, B. and M. Laguna (2006). "Fine-tuning of algorithms using fractional experimental designs and local search". In: *Operations Research* 54.1, pp. 99–114.

Ansótegui, C., Y. Malitsky, H. Samulowitz, M. Sellmann, and K. Tierney (2015). "Model-based genetic algorithms for algorithm configuration". In: *Proceedings of the 24th International Conference on Artificial Intelligence*. Buenos Aires, Argentina: AAAI Press, pp. 733–739.

Bartz-Beielstein, T., K. E. Parsopoulos, and M. N. Vrahatis (2004). "Design and analysis of optimization algorithms using computational statistics". In: *Applied Numerical Analysis & Computational Mathematics* 1.2, pp. 413–433.

Battiti, R. and M. Brunato (2005). *Reactive search: machine learning for memory-based heuristics*. Tech. rep. Teofilo F. Gonzalez (Ed.), Approximation Algorithms and Metaheuristics, Taylor & Francis Books (CRC Press.

Battiti, R. and G. Tecchiolli (1994). "The reactive tabu search". In: *ORSA journal on computing* 6.2, pp. 126–140.

Beasley, D., D. R. Bull, and R. R. Martin (1993). "An overview of genetic algorithms: Part 2, research topics". In: *University computing* 15.4, pp. 170–181.

Birattari, M. and J. Kacprzyk (2009). *Tuning metaheuristics: a machine learning perspective*. Vol. 197. Springer.

Birattari, M., Z. Yuan, P. Balaprakash, and T. Stützle (2010). "F-Race and iterated F-Race: an overview". In: *Experimental methods for the analysis of optimization algorithms*. Springer, pp. 311–336.

Boussaïd, I., J. Lepagnot, and P. Siarry (2013). "A survey on optimization metaheuristics". In: *Information Sciences* 237, pp. 82–117.

Bovet, D. P. and P. Crescenzi (1994). *Introduction to the theory of complexity*. Hertfordshire, UK: Prentice Hall International (UK) Ltd. ISBN: 0-13-915380-2.

Carvalho, A. R., F. M. Ramos, and A. A. Chaves (2011). "Metaheuristics for the feedforward artificial neural network architecture optimization problem". In: *Neural Computing and Applications* 20.8, pp. 1273–1284.

Clarke, G. and J. Wright (1964). "Scheduling of vehicles from a central depot to a number of delivering points". In: *Operations Research* 12, pp. 568–581.

Conover, W. J. (1999). *Practical nonparametric statistics*. 3rd ed. John Wiley & Sons.

Coy, S. P., B. L. Golden, G. C. Runger, and E. A. Wasil (2001). "Using experimental design to find effective parameter settings for heuristics". In: *Journal of Heuristics* 7.1, pp. 77–97.

Czarn, A., C. MacNish, K. Vijayan, B. Turlach, and R. Gupta (2004). "Statistical exploratory analysis of genetic algorithms". In: *Evolutionary Computation, IEEE Transactions on* 8.4, pp. 405–421.

De Jong, K. (2007). "Parameter setting in EAs: a 30 year perspective". In: *Parameter setting in evolutionary algorithms*. Springer, pp. 1–18.

Dobslaw, F. (2010). "A parameter tuning framework for metaheuristics based on design of experiments and artificial neural networks". In: *World Academy of Science, Engineering and Technology* 64, pp. 213–216.

Eiben, A. E., R. Hinterding, and Z. Michalewicz (1999). "Parameter control in evolutionary algorithms". In: *Evolutionary Computation, IEEE Transactions on* 3.2, pp. 124–141.

Gendreau, M., J.-Y. Potvin, G. Bräumlaysy O.and Hasle, and A. Løkketangen (2008). *Metaheuristics for the vehicle routing problem and its extensions: A categorized bibliography*. Springer.

Gunawan, A., H. C. Lau, and E. Wong (2013). "Real-world parameter tuning using factorial design with parameter decomposition". In: *Advances in Metaheuristics*. Springer, pp. 37–59.

Hastie, T., R. Tibshirani, and J. Friedman (2009). *The elements of statistical learning*. 2nd ed. Springer.

Hooker, J. N. (1995). "Testing heuristics: We have it all wrong". In: *Journal of Heuristics* 1.1, pp. 33–42.

Hutter, F., H. H. Hoos, K. Leyton-Brown, and T. Stützle (2009). "ParamILS: an automatic algorithm configuration framework". In: *Journal of Artificial Intelligence Research* 36.1, pp. 267–306.

Hutter, F., H. H. Hoos, and K. Leyton-Brown (2011). "Sequential model-based optimization for general algorithm configuration". In: *Learning and Intelligent Optimization*. Springer, pp. 507–523.

Jeong, S.-J., K.-S. Kim, and Y.-H. Lee (2009). "The efficient search method of simulated annealing using fuzzy logic controller". In: *Expert Systems with Applications* 36.3, pp. 7099–7103.

Johnson, D. S. (2002). "A theoretician's guide to the experimental analysis of algorithms". In: *Data structures, near neighbor searches, and methodology: fifth and sixth DIMACS implementation challenges* 59, pp. 215–250.

Juan, A. A., J. Faulin, J. Jorba, D. Riera, D. Masip, and B. Barrios (2011a). "On the use of Monte Carlo simulation, cache and splitting techniques to improve the Clarke and Wright savings heuristics". In: *Journal of the Operational Research Society* 62, pp. 1085–1097.

Juan, A. A., I. Pascual, D. Guimarans, and B. Barrios (2015c). "Combining biased randomization with iterated local search for solving the multidepot vehicle routing problem". In: *International Transactions in Operational Research* 22.4, pp. 647–667.

Lessmann, S., M. Caserta, and I. M. Arango (2011). "Tuning metaheuristics: a data mining based approach for particle swarm optimization". In: *Expert Systems with Applications* 38.10, pp. 12826–12838.

Maron, O. and A. W. Moore (1993). "Hoeffding races: accelerating model selection search for classification and function approximation". In: *Robotics Institute*, p. 263.

Martins, S. L. and C. C. Ribeiro (2006). "Metaheuristics and applications to optimization problems in telecommunications". In: *Handbook of optimization in telecommunications*. Springer, pp. 103–128.

Montero, E., M.-C. Riff, and B. Neveu (2014). "A beginner's guide to tuning methods". In: *Applied Soft Computing* 17, pp. 39–51.

Montgomery, D. C. (2008). *Design and analysis of experiments*. 8th ed. John Wiley & Sons.

Park, M.-W. and Y.-D. Kim (1998). "A systematic procedure for setting parameters in simulated annealing algorithms". In: *Computers & Operations Research* 25.3, pp. 207–217.

Pavón, R., F. Díaz, R. Laza, and V. Luzón (2009). "Automatic parameter tuning with a Bayesian case-based reasoning system. A case of study". In: *Expert Systems With Applications* 36.2, pp. 3407–3420.

Pongcharoen, P., W. Chainate, and P. Thapatsuwan (2007). "Exploration of genetic parameters and operators through travelling salesman problem". In: *Science Asia* 33.2, pp. 215–222.

Ramos, I. C. O., M. C. Goldbarg, E. G. Goldbarg, and A. D. D. Neto (2005). "Logistic regression for parameter tuning on an evolutionary algorithm". In: *Evolutionary Computation, 2005. The 2005 IEEE Congress on*. Vol. 2. IEEE, pp. 1061–1068.

Ridge, E. and D. Kudenko (2007). "Analyzing heuristic performance with response surface models: prediction, optimization and robustness". In: *Proceedings of the 9th annual conference on Genetic and evolutionary computation*. ACM, pp. 150–157.

Ries, J. (2009). "Instance-based flexible parameter tuning for meta-heuristics using fuzzy-logic". PhD thesis. University of Portsmouth.

Ries, J., P. Beullens, and D. Salt (2012). "Instance-specific multi-objective parameter tuning based on fuzzy logic". In: *European Journal of Operational Research* 218.2, pp. 305–315.

Rousseeuw, P. J. (1987). "Silhouettes: a graphical aid to the interpretation and validation of cluster analysis". In: *Journal of computational and applied mathematics* 20, pp. 53–65.

Smith, J. E. (2008). "Self-adaptation in evolutionary algorithms for combinatorial optimisation". In: *Adaptive and Multilevel Metaheuristics*. Springer, pp. 31–57.

Talbi, E.-G. (2009). *Metaheuristics: From design to implementation*. New Jersey: John Wiley & Sons.

Theodoridis, S. and K. Koutroumbas (2009). *Pattern recognition*. Vol. 74. John Wiley & Sons.

Viana, A., J. P. Sousa, and M. A. Matos (2005). "Constraint oriented neighbourhoods – A new search strategy in metaheuristics". In: *Metaheuristics: progress as real problem solvers*. Springer, pp. 389–414.

Xu, J., S. Y. Chiu, and F. Glover (1998). "Fine-tuning a tabu search algorithm with statistical tests". In: *International Transactions in Operational Research* 5.3, pp. 233–244.

Zennaki, M. and A. Ech-Cherif (2010). "A new machine learning based approach for tuning metaheuristics for the solution of hard combinatorial Optimization problems". In: *Journal of Applied Sciences* 10, pp. 1991–2000.

A.4 Waste collection under uncertainty: A simheuristic based on variable neigborhood search

Aljoscha Gruler[1], Carlos L. Quintero-Araujo[1,2], Laura Calvet[1], Angel A. Juan[1]

1. IN3 – Computer Science, Multimedia and Telecommunications Department
Open University of Catalonia
Castelldefels, Spain
2. International School of Economics and Administrative Sciences
Universidad de La Sabana
Chía, Colombia

e-mail: {agruler, cquinteroa, lcalvetl, ajuanp}@uoc.edu

Abstract

Ongoing population growth in cities and increasing waste production has made the optimization of urban waste management a critical task for local governments. Route planning in waste collection can be formulated as an extended version of the well-known Vehicle Routing Problem, for which a wide range of solution methods already exist. Despite the fact that real-life applications are characterized by high uncertainty levels, most works on waste collection assume deterministic inputs. In order to partially close this literature gap, this paper first proposes a competitive metaheuristic algorithm based on a Variable Neighborhood Search framework for the deterministic Waste Collection Problem. Then, this metaheuristic is extended to a simheuristic algorithm in order to deal with the stochastic problem version. This extension is achieved by integrating simulation into the metaheuristic framework, which also allows a closer risk analysis of the best-found stochastic solutions. Different computational experiments illustrate the potential of our methodology.

Keywords: waste collection management, vehicle routing problem, metaheuristics, simulation, risk analysis, stochastic optimization, simheuristics.

1. Introduction

In the face of rising population densities in urban areas around the world, a large number of cities are currently reorganizing their municipal responsibilities (Nations, 2015). In this context, solid waste management is arguably "the most important municipal service" a city provides for its residents (The World Bank, 2012). A number of strategic, tactical, and operational issues –for example related to the location of disposal sites or landfills, clustering of service territory, vehicle routing, etc.– need to be addressed (Ghiani et al., 2014a). Especially the collection phase is highly important. On the one hand, uncollected garbage can lead to pollution of the environment and health-issues, while noise and road congestions through extensive use of waste collection vehicles decrease urban living standards. On the other hand, waste collection represents up to two thirds of operational waste management costs (Malakahmad et al., 2014; Son, 2014; Tavares et al., 2009). Consequently, the Waste Collection Problem (WCP) for effective route planning in municipal waste collection is of high practical importance, especially in the context of smart city initiatives (Neirotti et al., 2014).

Typically, the WCP is either modeled as a rich Vehicle Routing Problem (VRP) (Toth and Vigo, 2014; Caceres et al., 2014) or as an Arc Routing Problem (ARP) (Corberán and Laporte, 2014), depending on the type of waste to be collected. While the collection of household refuse in small bins from private homes is often modeled as an ARP, the VRP as node routing model is more suitable for the collection of waste from larger containers, which are often located close to retailers, construction sites, or waste collection points of building blocks in metropolitan areas. This work addresses the WCP as a VRP in the following. The reader is referred to Ghiani et al. (2014b) and Han and Ponce-Cueto (2015) for a more detailed discussion between both modeling types.

Extending the classical VRP formulation, the WCP consists of a set of waste containers (customers) with associated waste levels (demands) and a central depot in which a capacitated vehicle fleet is located. Furthermore, there is a set of landfills at which collected waste is disposed. The

arcs (edges) connecting any two nodes are characterized by travel costs, e.g.: distance, time, or CO_2 emissions. Figure A.1 illustrates an example of a WCP solution with two routes. Vehicles start at the central depot to visit a number of waste containers. A WCP specific problem constraint is that vehicles start and end their routes empty. For this reason, at least one additional landfill trip is included on every route before the collection vehicle returns to the central depot. As can be seen in Route 2, multiple landfill visits during the same trip are also possible. Thus, a vehicle might visit a disposal site once its capacity is reached and then continue the same trip as long as no further route constraints (e.g., time windows, maximum number of stops, etc.) are violated.

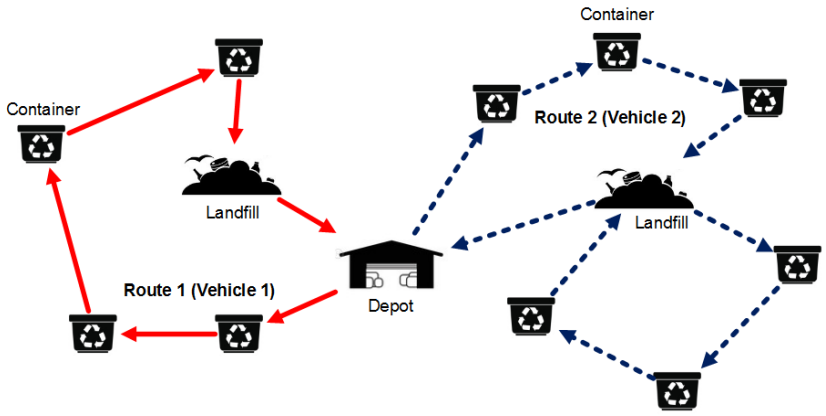


FIGURE A.1: Example of a 2-route solution for a simple WCP instance.

In many real-world applications of routing problems, uncertainty is one of the major factors to be considered during operational planning. Typically, relevant information and input data is not perfectly available. While uncertainty can practically influence any variable, the most considered cases of stochasticity include: *(i)* customer demands; *(ii)* travel times; and *(iii)* stochastic customers (Gendreau et al., 2014). For a more detailed overview on stochastic optimization problems and available solution approaches, see Bianchi et al. (2006), Bianchi et al. (2009), and Pillac et al. (2013), or Ritzinger et al. (2016). A typical application area experiencing high levels of uncertainty is urban waste collection (Beliën et al. (2014)). As waste generation and travel times of vehicles cannot be predicted with full certainty, there is a need for fast and risk-aware solutions of high quality which are able to take stochastic input variables into account. Accordingly, this paper presents a simheuristic methodology (Juan et al., 2015a) that combines a metaheuristic algorithm with simulation techniques to address the WCP with stochastic waste levels.

Given a WCP with uncertain inputs, our methodology transforms the stochastic problem instance into its deterministic counterpart by using expected values. Then, a competitive metaheuristic for the deterministic WCP is employed to obtain a set of promising *a priori* solutions, each defining a certain order of nodes to visit and the associated deterministic costs. As the actual amount of waste to be collected at each container is only revealed once reached by the vehicle, the predefined route is subject to the risk of route failures, where vehicles reach their capacities before the planned landfill stop. In this case, an additional and costly landfill trip is necessary. For this reason, Monte Carlo simulation (MCS) is employed to simulate different waste levels and assess the performance of the most promising deterministic solutions to different stochastic scenarios. Therefore, our approach reaches high-quality solutions to large-sized problem instances with up to 2100 nodes in only a few minutes for both the deterministic and stochastic cases. The contributions of this paper are threefold: *(i)* a competitive Variable Neighborhood Search (VNS) metaheuristic is presented, outperforming the current best known solutions of a well-known benchmark set for the deterministic WCP; *(ii)* the metaheuristic is extended using MCS to allow a fast solution to realistic WCP problems experiencing uncertainty regarding waste levels; and *(iii)* a risk-analysis of the generated solutions based on the expected reliability according to different waste level variances and vehicle capacity safety levels is performed.

This paper is structured as follows: a literature review on related work is provided in Section 2; the WCP is defined in Section 3; our approach to the deterministic WCP is outlined in Section 4, before Section 5 discusses its simheuristic extension of the stochastic version; in Section 6 a set of computational experiments conducted to validate our approach based on large-sized WCP

instances with stochastic input variables is described and analyzed; and finally, the conclusions of this work and possible lines of future research are discussed in Section 7.

2. Related Work

Probably the first work to address municipal solid waste collection was introduced by Beltrami and Bodin (1974). Since then, various solution techniques for different variants of the WCP and its extensions have been proposed. While some works formulating the WCP as an ARP can be found (Ghiani et al. (2005) and Bautista et al. (2008)), the following discussion focuses on recent publications using VRP formulations. More extensive literature reviews are provided by **Golden14**; Beliën et al. (2014) and Ghiani et al. (2014b) and Han and Ponce-Cueto (2015).

The deterministic Waste Collection Problem

Most works on the deterministic WCP focus on case studies with some problem extension, e.g.: combined routing and vehicle scheduling. For example, Baptista et al. (2002) elaborated an extension of the Christofides and Beasley heuristic for the multi-period WCP (Christofides and Beasley, 1984), modeled as a periodic VRP (PVRP) to combine vehicle scheduling over multiple time periods with route planning. The authors used their approach to improve municipal waste collection in a Portuguese city. Also addressing a multi-period WCP, Teixeira et al. (2004) developed a cluster-first route-second heuristic to schedule and plan waste collection routes for different waste types in a case study in Portugal with over 1600 collection sites. Nuortio et al. (2006) presented a guided variable thresholding metaheuristic to solve a multi-period WCP with several thousand collection points in Eastern Finland. Hemmelmayr et al. (2013) addressed the PVRP with different waste types and up to 288 containers, which they solved with a VNS metaheuristic. They consider the landfills as intermediate facilities, which are inserted in pre-constructed routes using dynamic programming. In the same work, the authors also discussed the single period WCP with multiple depots, in which the landfills serve as vehicle depots and disposal sites at the same time. Later, Hemmelmayr et al. (2014) discussed the integrated vehicle routing- and bin allocation problem using the same real-life problem set, which they solved with a combination of a VNS metaheuristic for the routing part and a mixed integer linear programming-based exact method for the bin allocation. Ramos et al. (2014) extended the typical objective of minimizing routing costs in order to include environmental concerns, considering multiple waste types and numerous vehicle depots in a case study in Portugal.

Only focusing on waste collection routing, Kim et al. (2006) developed an extension of Solomon's insertion algorithm (Solomon, 1987) to optimize routes of a North American waste management service provider, considering a capacitated vehicle fleet, time windows, and driver lunch breaks. The authors reported reduced routing distances of up to 10%. Furthermore, a benchmark set of 10 realistic instances based on the original case study ranging from 102-2100 nodes is provided. This benchmark set was later employed by Ombuki-Berman et al. (2007) to test a multi-objective Genetic Algorithm. Furthermore, the same benchmark set was used by Benjamin and Beasley (2010) and Buhrkal et al. (2012) to test their metaheuristic solution methods. Benjamin and Beasley (2010) combined Tabu Search with VNS. By exchanging containers and landfills within and between routes, the solution search space is systematically increased. Buhrkal et al. (2012) put forward an Adaptive Large Neighborhood Search metaheuristic. Based on an initial solution, this approach applies a range of destroy-and-repair methods to examine several solution neighborhoods. It is called adaptive since the choice of methods depends on the solution quality obtained during the construction of earlier solutions. Moreover, an acceptance criterion for new solutions based on Simulated Annealing is included. Recently, Markov et al. (2016) presented a multiple neighborhood search heuristic for a real-word application of the waste collection VRP with intermediate facilities. The authors consider a heterogeneous vehicle fleet and flexible depot destinations in their approach.

The stochastic Waste Collection Problem

Concerning the WCP with stochastic demands, the literature is more scarce. Ant Colony Optimization and a hybrid approach based on a Genetic Algorithm and Tabu Search for a case study with 50 containers in Malaysia is presented in Ismail and Irhamah (2008) and Ismail and Loh

(2009). After planning a priori routes, waste levels are simulated according to a discrete probability distribution. Routes undergo a recourse action (i.e., an additional disposal trip) whenever actual demand exceeds the planned collection amount. Nolz et al. (2014) formulated a collector-managed inventory routing problem for a case study on the collection of infectious waste. By using real information obtained through radio frequency identification, their Adaptive Large Neighborhood Search algorithm is able to consider stochastic inputs. Alshraideh and Abu Qdais (2016) combined a multi-period WCP with time windows and stochastic demands in an areal case study of medical waste collection from 19 hospitals in Northern Jordan. They used a Genetic Algorithm and a probability constraint regarding a pre-defined service level to solve the problem.

Related to the dynamic WCP in combination with modern Information- and Communication Technology (ICT), Johansson (2006) tested the introduction of volumetric sensors in Malmoe (Sweden) through analytic modeling and discrete-event simulation. They compare static routing with its dynamic counterpart in which containers raise alarms when a certain waste level is exceeded, suggesting that dynamic waste collection routing could lead to improved operations. Later, Faccio et al. (2011) elaborated a real-time tractability system for multi-objective waste collection in an Italian city. They discuss the use of ICT (e.g., volumetric sensors, Radio Frequency Identification, GPS, etc.) to connect containers, vehicles, and the operations center in an economic feasibility analysis. More related to simulation than optimization, Wang (2001) developed an integrated simulation model for solid waste collection with both deterministic and stochastic waste generation- and household set-out rates. Their decision support tool can be used to evaluate collection systems concerning environmental and operational costs and optimize the system design under different circumstances. Also related to stochastic waste collection, Yeomans (2007) integrated grey programming into evolutionary simulation-optimization to solve solid waste collection problems with high levels of uncertainty. Gruler et al. (2015a) describe the combination of a multi-start algorithm with simulation to solve the WCP with stochastic demands. While they propose some generic ideas on which our approach is built, their paper only provides a general overview of a basic solving approach and no computational experiments were run.

Regarding simheuristic methodologies for tackling routing problems under uncertainty, Juan et al. (2011b) addressed the VRP with stochastic demands, analyzing the effect of safety stocks concerning total routing costs and solution reliability. Gonzalez et al. (2016) presented a simheuristic algorithm to deal with the ARP with stochastic demands. The Inventory Routing Problem with stock-outs and stochastic demands was discussed by Juan et al. (2014b), outlining the effect of different refill strategies on inventory and routing costs in a two-echelon supply chain.

3. Description of the problem

As a rich extension of the VRP, the WCP is NP-hard (Caceres et al., 2014). To start with, this Section defines a basic version of the problem, including the mathematical formulation and a small computational experiment justifying the need of a metaheuristic-based approach. Afterwards, a richer and more realistic version is presented.

Basic version of the WCP

The WCP can be described on a graph $G = (V, A)$, where the set of nodes $V = V^d \cup V^f \cup V^c \cup V^b$ includes: *(i)* a set of starting and ending depots $V^d = \{0, 0'\}$ (in practice both depots could be the same), with the starting depot being the initial location of a fleet of homogeneous vehicles $K = \{1, 2, \ldots, k\}$, each of them having a capacity C; *(ii)* a set $V^f = \{1, 2, \ldots, m\}$ describing m landfills at which collected waste must be disposed at least once before visiting the ending depot (see Figure A.1); *(iii)* a set of waste containers (customers) $V^c = \{m+1, \ldots, m+n\}$ with associated waste levels $q_i > 0$ ($\forall i \in V^c$); and *(iv)* a set $V^b = \{0^*\}$ representing a virtual lunch-break node that has to be included in each route. Each node $i \in V \setminus V^d$ has an associated time window represented by $[a_i, b_i]$ (with $0 \leq a_i < b_i$). Necessary service times for emptying any container and the duration of the lunch break are formulated as $r_i > 0$ ($\forall i \in V^c \cup V^b$). Likewise, the set $A = \{(i, j)/i, j \in V, i \neq j\}$ describes the arcs connecting any pair of different nodes. Each pair is characterized by its respective travel costs, $c_{ij} = c_{ji} \geq 0$, and travel times, $t_{ij} = t_{ji} \geq 0$. The travel time associated with going from any node $i \in V \cup V^b$ to the virtual lunch-break node (and vice versa) is equal to zero, i.e.: $t_{i0^*} = t_{0^*i} = 0$. Notice, however, that the travel cost associated with 'crossing' the lunch-break virtual node is given by the travel cost of the origin and destination

nodes, i.e.: $c_{i0^*} + c_{0^*j} = c_{ij}$. The decision variables x_{ijl} ($\forall (i,j) \in A$, $\forall l \in K$) equal 1 if arc (i,j) is employed by vehicle l and 0 otherwise. Our mathematical model is presented next. It extends the one proposed in Buhrkal et al. (2012) (e.g. by including the lunch break constraints) and the one proposed in Sahoo et al. (2005) (which only considers traveling times). In our model, d_{il} represents the accumulated load of vehicle l before serving node i, h_{il} represents the service starting time of vehicle l at node i, and M_1 is a large-enough constant that can be defined as $M_1 = \max\{b_i\}(\forall i \in V \setminus V^d) + \max\{s_i\}(\forall i \in V \setminus V^d) + \max\{t_{ij}\}\ (\forall (i,j) \in A)$.

$$\text{Min} \sum_{(i,j)\in A} c_{ij} \sum_{l\in K} x_{ijl} \tag{A.1}$$

Subject to:

$$\sum_{j\in V\setminus V^d} x_{0jl} = 1 \qquad \forall l \in K \tag{A.2}$$

$$\sum_{i\in V^f} x_{i0'l} = 1 \qquad \forall l \in K \tag{A.3}$$

$$\sum_{i\in V} \sum_{l\in K} x_{ijl} = 1 \qquad \forall j \in V^c \tag{A.4}$$

$$\sum_{\substack{i\in V \\ i\neq j}} x_{ijl} = \sum_{\substack{i\in V \\ i\neq j}} x_{jil} \qquad \forall j \in V \setminus V^d, \forall l \in K \tag{A.5}$$

$$a_i \leq h_{il} \leq b_i \qquad \forall i \in V \setminus V^d, \forall l \in K \tag{A.6}$$

$$h_{il} + r_i + t_{ij} \leq h_{jl} + (1 - x_{ijl})M_1 \qquad \forall (i,j) \in A, \forall l \in K \tag{A.7}$$

$$d_{il} = 0 \qquad \forall i \in V^d, \forall l \in K \tag{A.8}$$

$$d_{jl} - C(1 - x_{ijl}) \leq d_{il} + q_i \leq d_{jl} + C(1 - x_{ijl}) \qquad \forall i \in V^c \cup V^b \cup \{0\}, \forall j \in V \setminus V^d, \forall l \in K \tag{A.9}$$

$$d_{jl} \leq C(1 - x_{ijl}) \qquad \forall i \in V^f, \forall j \in V^c \cup V^b, \forall l \in K \tag{A.10}$$

$$\sum_{i\in V} x_{i0^*l} = 1 \qquad \forall l \in K \tag{A.11}$$

$$\sum_{j\in V} x_{0^*jl} = 1 \qquad \forall l \in K \tag{A.12}$$

$$h_{il} + r_i + r_{0^*} + t_{ij} \leq h_{jl} + (2 - x_{i0^*l} - x_{0^*jl})(M_1 + r_{0^*}) \qquad \forall (i,j) \in A, \forall l \in K \tag{A.13}$$

$$d_{il} \leq C \qquad \forall i \in V^f, \forall l \in K \tag{A.14}$$

$$c_{i0^*} + c_{0^*j} \geq c_{ij} \qquad \forall (i,j) \in A \tag{A.15}$$

$$d_{il} \geq 0 \qquad \forall i \in V, \forall l \in K \tag{A.16}$$

$$x_{ijl} \in \{0,1\} \qquad \forall (i,j) \in A, \forall l \in K \tag{A.17}$$

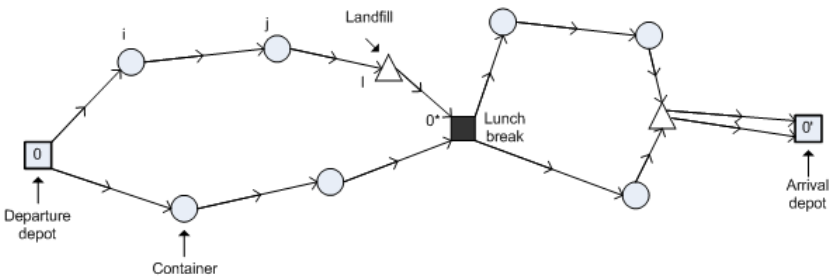


FIGURE A.2: Representation of the WCP.

The objective function of minimizing total cost is formulated in Equation (1), which represents the costs of the edges selected (including the ones 'crossing' a lunch-break node). Constraints (2) imply that each vehicle leaves the starting depot, while constraints (3) impose that each vehicle

must visit a landfill right before reaching the ending depot. Constraints (4) ensure that each container is visited exactly once. Constraints (5) guarantee that inflow and outflow to each non-depot node must be equal. Constraints (6) force to the compliance of time windows. Constraints (7) define the earliest possible starting time for the next customer taking into account service and travel times. Constraints (8) reset the vehicle load to zero when leaving from or arriving at a depot. Constraints (9) take care of the accumulating load levels after visiting each container. Constraints (10) force that vehicles are empty after a visit to a landfill. Constraints (11) and (12) introduce a lunch break during each route. Constraints (13) impose that travel times between the stops before and after the lunch break are taken into account to fix the earliest possible starting time of the next container. Constraints (14) limit the maximum waste a vehicle may carry at any time. Constraints (15) define the costs of crossing a virtual lunch-break node. Notice that these costs are calculated as the travel cost between the origin- and destination node, i.e.: $c_{i0^*} + c_{0^*j} = c_{ij}$. Thus, c_{i0^*} and c_{0^*j} are not inputs but decision variables satisfying the aforementioned constraint. Finally, constraints (16) and (17) define variable domains.

The previous model was implemented in the GAMS language (Version 23.5.2). Then, the CPLEX solver (Version 12.2.0.0) was used to try solving the smallest instance provided by Kim et al. (2006), which has 1 depot, 99 containers, and 2 landfills. However, the solver ran out of memory after 54 minutes of computation. Therefore, we generated three smaller instances with 20, 24, and 44 containers, respectively. The number of landfills used was 2, as in the original instances. The CPLEX solver was allowed to run for a maximum time of 48 hours or until a gap lower than 1% was reached. Then, we also employed our VNS algorithm –described in future sections– to solve the same instances. Table A.1 shows the comparison of results between CPLEX and our VNS algorithm for the aforementioned instance. For each solving method, we include the best solution found (Z), the time consumed to find that solution ($TC\ Z$) and the maximum computing time allowed(TC). Notice that both methods provide optimal solutions for the first two instances, but the VNS clearly outperforms the exact method in computing times (less than 1 second compared to 126 and 854 seconds required by CPLEX, respectively). Regarding the third instance, CPLEX ran out of memory after 5864 seconds. The best solution found by the exact method –after 2306 seconds– has an objective value of 70.85 (relative gap of 65% with respect to the lower bound), while our VNS algorithm provides an objective value of 63.09 in 1.5 seconds. These results reveal how difficult it becomes for the CPLEX solver to find optimal/near-optimal solutions in low computing times, even for small instances of the basic WCP version. For that reason, in the following sections we will focus on developing heuristic-based approaches, which allow us to deal with richer and more realistic versions of the problem.

Table A.1: Comparison of results among CPLEX and our VNS

	CPLEX			**VNS**			
Instances	Z	*TC Z (sec.)*	*TC (sec.)*	Z	*TC Z (sec.)*	*TC(sec.)*	*GAP*
Kim102(20)	38.19	126.31	8916	38.19	<1	300	0.00%
Kim102(24)	24.88	854.05	3641.51	24.88	<1	300	0.00%
Kim102(44)	70.85	2306.55	5864.28*	63.09	1.5	300	-10.95%
				Average			*-3.65%*

A richer and more realistic version of the WCP

The following restrictions, which significantly increase the difficulty of the problem, are added to the basic version described before: *(i)* the number of vehicles used is not predetermined, only the maximum number of available vehicles is given; *(ii)* the lunch break is automatically included in a route whenever a certain time window is reached; *(iii)* there is a maximum number of stops at containers and landfills per route; *(iv)* there is a maximum amount of waste that can be collected on a single vehicle route; and *(v)* the depot also has a time window. In the next sections of this paper, we will develop methodologies for both the deterministic and the stochastic versions of this rich WCP.

4. A VNS metaheuristic for the deterministic Waste Collection Problem

In order to solve the deterministic WCP we propose a VNS metaheuristic, which is based on the construction of different solution neighborhoods and the following descent phase to define a local

minimum in the corresponding neighborhood structure (Hansen et al., 2010b). An initial solution is obtained by applying the well-known savings routing heuristic (Clarke and Wright, 1964) and its biased-randomized extension as described in Faulin et al. (2008) and in Juan et al. (2013a). This procedure is adapted to the special case of waste collection by changing the calculation of savings values used for merging two customers i and j, originally calculated as $s_{ij} = c_{i0} + s_{0j} - c_{ij}$ (Figure A.3 - left). In the WCP, the costs of traveling between a customer and the depot are asymmetric due to the additional landfill visit. To address this new situation, we employ a simple transformation based on the average savings associated to each arc (Figure A.3 - right).

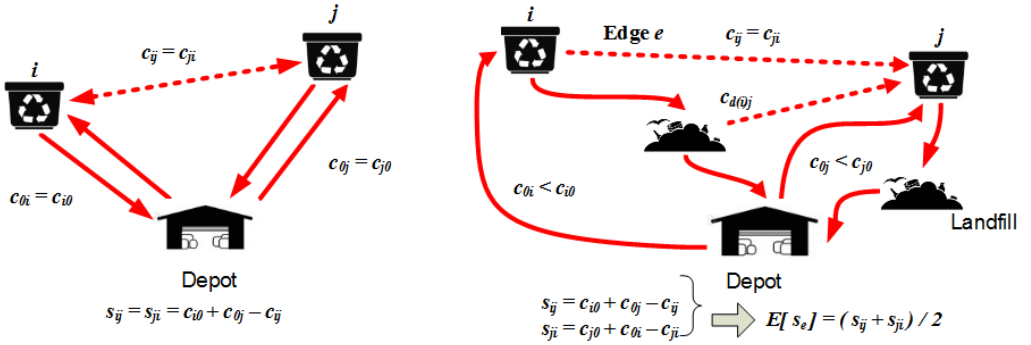


FIGURE A.3: Savings of the original CWS heuristic (left) and expected savings proposed for the WCP (right)

Based on the initial solution $baseSol$, different neighborhood structures $N_k(k = 1, ..., k_{max})$ are created. The shaking procedures applied in this work to create new solution structures are outlined in Table A.2. Within each neighborhood $N_k(baseSol)$, different local descent heuristics described in Table A.3 are randomly applied to find the local minimum of $N_k(baseSol)$. To conclude the local search phase, a quick solution improvement procedure based on a cache memory technique (Juan et al., 2013a) is implemented: the best-known order of traveling between a set of nodes establishing a sub-route –i.e., starting at the depot or a landfill and ending at a disposal site– is stored in a hash-table data structure, thus allowing new solutions to benefit from previously constructed ones. Whenever the local search phase leads to a more competitive objective function value than that of $baseSol$, $baseSol$ is updated and k is returned to its initial value of 1. If $baseSol$ cannot be improved through the local minimum of N_k, k is incremented by 1 and the next shaking operator is applied. Once each neighborhood has been constructed ($k = k_{max}$), the process is repeated until a certain predefined stopping criterion (e.g.: time, iterations, etc.) has been reached. Note that we shuffle the list of neighborhood operators every time $k > k_{max}$. A description of the VNS procedure for the deterministic WCP can be seen in Algorithm 1.

TABLE A.2: Shaking operators

Operator (k)	Description
Customer Swap Inter-Route	Swaps two random customers between different routes.
2-Opt Inter-Route	Interchanges two chains of randomly selected customers between different routes.
Reinsertion Inter-Route	Inserts a random customer in a different route.
Cross-Exchange	Interchanges positions of 2-4 random, non-consecutive customers from different routes.

TABLE A.3: Local Search operators

Operator (LS-Scheme)	Description
Best Position Insertion	Reinserts the container with the highest objective function increase into the best available position of any route.
Re-allocate all	Iteratively calculates the objective function increase of each container and reinserts it at the best possible position.
Random Swaps	Randomly selects and interchanges two nodes (from the same or different routes) if the objective function improves.

Algorithm 1

```
1: baseSol ← solve biased randomized CWS for the WCP                ▷ Juan et al. (2013a)
2: while stopping criteria not reached do
3:     shuffle(ListOfShakingOperators)
4:     k ← 1
5:     repeat
6:     newSol ← shake(baseSol, k)                                   ▷ see Table A.2
7:     improving ← true                                             ▷ Start Local Search
8:     while improving do
9:         newSol* ← localDescent(newSol, randomLSoperator)
                                                                    ▷ see Table A.3
10:        if costs(newSol*) ≤ costs(newSol) then
11:            newSol ← newSol*
12:        else
13:            improving ← false
14:        end if
15:        cacheSubRoutes(newSol)                                   ▷ End Local Search
16:        if costs(newSol) < costs(baseSol) then
17:            baseSol ← newSol
18:            k ← 1
19:        else
20:            k ← k + 1
21:        end if
22:    end while
23:    until k > k_max
24: end while
25: bestSol ← baseSol
26: return bestSol
```

To test the competitiveness of our algorithm we use the benchmark instances provided by Kim et al. (2006), which were later also adopted by Benjamin and Beasley (2010) and Buhrkal et al. (2012) to validate their solution approaches for the WCP. This benchmark set includes 10 realistic instances, ranging from 102-2100 nodes with time windows, multiple landfills, a single depot, a driver lunch break during each route, and a homogeneous vehicle fleet. Furthermore, we compare our approach to the clustered instances presented by Buhrkal et al. (2012). A clustering procedure is applied to nodes with the same location and time windows to change the total number of nodes. The algorithm was implemented as Java application and run on a personal computer with an Intel®Xeon™CPU E5-2630 v2 @ 2.60GHz processor. The initial solutions constructed with the biased randomized version of the savings heuristic are based on a distribution parameter randomly chosen within the range (0.4, 0.5) at each solution construction step.

The results are summarized in Table A.4. Column (1) reports the best known solution (BKS) for each instance (listed as Kim_numberOfNodes) as reported in the works of Kim et al. (2006), Benjamin and Beasley (2010), and Buhrkal et al. (2012). The computational times (CT) in seconds, to reach each solution can be seen in column (2), while column (3) lists the average results with 10 different random number seeds as presented in the benchmark papers. Notice that the benchmark papers use different computers, computational times, and programming languages to implement and execute their described algorithms, making a fair comparison difficult. For this reason, we have tested our VNS metaheuristic with two different stopping criteria. On the one hand, our best solution (achieved with 10 different random number seeds) when applying the CTs listed in column (2) is reported in column (4). Furthermore, we report our average solution with 10 different random number seeds (5) and our best solution (6) with a stopping criterion of 300 seconds per instance as suggested by Benjamin and Beasley (2010). It can be seen that our algorithm outperforms current BKS's by an average of -0.85% and -2.65%. Moreover, our algorithm reaches 9 new BKS's (11 with the extended algorithm running time). As can be observed, the percentage gap compared to the BKS extends to more than 10% in some cases. These differences are supported by results described in a technical report by Markov et al. (2015), in which the authors use the five smallest (non-clustered) instances of the applied benchmark set to test a heuristic for

TABLE A.4: Computational results for the deterministic case and comparison with BKSs

Instance	(1) BKS	(2) CT BKS (s)	(3) BKS average	(4) Our best sol[1]	(5) Our sol average[2]	(6) Our best sol[2]	(7) CT our best sol (s)	%-Gap (1)-(4)	%-Gap (1)-(6)
Kim102	174.5	3	176.03	158.61	158.64	154.62	5	-9.11	-11.39
Kim277	447.6	8	455.7	472.73	457.14	450.6	299	5.61	0.67
Kim335	182.1	10	196.49	189.79	187.36	184.22	298	4.22	1.16
Kim444	78.3	18	78.99	80.22	80.09	79.49	292	2.45	1.52
Kim804	604.1	72	650.65	603.17	601.14	593.2	300	-0.15	-1.80
Kim1051	2250.6	194	2387.7	2128.37	2119.50	2077.37	294	-5.43	-7.70
Kim1351	871.9	105	891.17	929.5	929.40	910.6	238	6.61	4.44
Kim1599	1337.5	252	1385.3	1184.67	1208.54	1182.58	292	-11.43	-11.58
Kim1932	1162.5	285	1192.2	1149.45	1169.95	1136.34	273	-1.12	-2.25
Kim2100	1749	356	1916.8	1595.48	1622.29	1603.93	293	-8.78	-8.29
Clustered Instances									
Kim86	174.5	3	176.6	155.68	158.35	155.68	10	-10.79	-10.79
Kim267	450.7	8	456.4	460.4	455.96	449.41	294	2.15	-0.29
Kim322	182.4	10	190.7	189.78	185.93	184.26	298	4.05	1.02
Kim444	78.6	18	79.2	80.22	80.09	79.49	292	2.06	1.13
Kim602	586.2	72	647.8	610.52	593.25	586.11	297	4.15	-0.02
Kim1011	2295.2	116	2370.5	2151.51	2131.00	2102.23	299	-6.26	-8.41
Kim536	850	105	850.9	885.83	877.69	850.46	292	4.22	0.05
Kim870	1170.2	252	1230.6	1156.15	1180.07	1145.83	286	-1.20	-2.08
Kim1860	1128.7	285	1180.9	1129.89	1154.48	1138.6	295	0.11	0.88
Kim1877	1594.2	266	1650.8	1620.89	1642.20	1604.33	186	1.67	0.64
Average	***868.44***	***122***	***908.27***	***846.64***	***849.65***	***833.47***	***257***	***-0.85***	***-2.65***

[1]computational times per instance equal to column (2)
[2]computational times per instance equal to column (7)

the WCP.

Some final remarks concerning the algorithm can be made. The initial solution for all instances is constructed in under 3 seconds (only a few milliseconds for the smaller problem cases). In comparison to the previous BKS's, the average gap of the initial solutions is 8.92%. A similar comparison to our best solution is done with the different local search operators. When only running the algorithm with the "best position insertion"-, the "re-allocate all"-, and the "random-swaps" local search, the average percentage gaps are -0.38%, 1.28%, and 2.34% respectively. While performance differences between the operators can be observed, these results suggest that the combination of various local search techniques is useful in the solution of the WCP.

5. Solving the stochastic Waste Collection Problem

In contrast to deterministic cases of the WCP, waste levels cannot be predicted with full certainty when solving a more realistic stochastic version of the problem. The fact that actual waste levels in containers are only known when reaching designated pick-up points can lead to route failures whenever collected garbage exceeds the planned collection amount. In these cases, the collection vehicle needs to add an additional and expensive landfill visit to its route. The proposed simheuristic methodology outlined in Algorithm 2 allows an estimation of the solution quality of previously created outputs using the VNS metaheuristic proposed in Section 4 by integrating MCS into the solution procedure. Note that the simheuristic structure for the WCP can theoretically be combined with any metaheuristic approach addressing the problem setting. However, the quality of the stochastic solution is directly related to the results obtained in the deterministic metaheuristic process (Juan et al., 2015a). For this reason, the use of an efficient deterministic solution process such as the one outlined in Section 4 is beneficial.

Before simulating waste levels, our methodology starts by transforming the stochastic input variables into their deterministic counterpart, which is used to establish initial WCP solutions. Even though waste levels (especially in urban settings) face different levels of stochasticity, their behavior can typically be modeled according to some kind of theoretical or empirical distribution (e.g., based on historical data). This allows the (stochastic) waste levels w_i at each container i to be replaced with expected values $E[w_i]$. Using these deterministic values, an initial solution $baseSol$ is constructed. In the following, the solution quality in a stochastic environment is tested by randomly simulating the waste levels of each container i for a certain number of iterations

(or simulation runs) within the predefined probability distribution. During each run the occurring route failure costs are estimated by penalizing situations in which vehicle capacities are reached before a scheduled landfill trip. More specifically, route failure costs are calculated as corrective actions to the predefined routes –i.e., the necessary additional landfill trip starting and ending at the container at which the vehicle capacities are reached. Finally, the sum of all route failure costs of all simulation runs are divided by the number of simulation runs. Thus, the expected total costs of *baseSol* now consist not only of the deterministic routing costs, but rather in the addition of the deterministic routing costs with the expected route failure costs. At this stage we propose the application of a small number of iterations *shortSimIter*. On the one hand, a larger number of simulation runs lead to more reliable estimates of the stochastic route costs. On the other hand, at this stage a shorter simulation procedure can be used to keep the computational effort through the simulation reasonable.

Algorithm 2

```
1:  replace stochastic waste levels by expected values          ▷ Creation of det. inputs
2:  baseSol ← solve biased randomized CWS for the WCP
3:  shortSimulation(baseSol)                                    ▷ MCS
4:  while stopping criteria not reached do
5:      k ← 1
6:      repeat
7:          newSol ← shake(baseSol, k)                          ▷ see Algorithm 1
8:          localSearch(newSol)                                 ▷ see Algorithm 1
9:          if detCosts(newSol) < detCosts(baseSol) then        ▷ Solution is promising
10:             shortSimulation(newSol)                         ▷ MCS
11:             if totalCosts(newSol) < totalCosts(baseSol) then
12:                 update(eliteSols)
13:                 baseSol ← newSol
14:                 k ← 1
15:             else
16:                 k ← k + 1
17:             end if
18:         end if
19:     until k > k_max
20: end while
21: for each eliteSol do
22:     longSimulation(eliteSol)
23:     estimateReliability(eliteSol)
24: end for
```

Once $detCosts(baseSol)$, $stochCosts(baseSol)$, and $totalCosts(baseSol)$ have been defined, new deterministic solution neighborhoods are constructed and locally improved as described previously. A newly constructed solution *newSol* is considered as promising whenever it yields lower deterministic costs than the current base solution. The behavior of each promising solution under waste level uncertainty is then evaluated by applying a short simulation run, leading to a first estimation of the total solution costs. Whenever $totalCosts(newSol) < totalCosts(baseSol)$, the current base solution is updated and k is returned to its initial value. Furthermore, the solution is stored as elite stochastic solution. With each elite solution, a more extensive simulation run is started for *longSimIter* iterations once the metaheuristic stopping criteria has been reached. As discussed in Juan et al. (2015a), it is recommendable to use a restricted number of solutions for the more extensive simulation run at this stage. For this reason, we limit the number of stored *eliteSols* to a maximum of 10. While some changes in the stochastic objective function of single solutions can be observed through the more detailed simulation, an augmented elite solution list has not shown any significant changes in the final ranking of the best stochastic solutions.

In addition to calculating the stochastic objective function value of promising deterministic solutions, our methodology allows the estimation of a solution reliability by considering the proportion of runs where the solution plan can be implemented without any route failure (a route failure occurs whenever the actual demand at any container exceeds the vehicle capacity, which forces the vehicle to visit a disposal site before resuming the original route). Thus, the

reliability $reliab_r$ of each route r of any solution S is computed as the quotient of the number of runs in which a route failure occurs divided by the total number of simulation runs, i.e. $reliab_r = simRunsWithRouteFailue/simRuns$. Notice that each route in a solution can be seen as an independent component of a series system (i.e., the proposed solution will fail if, and only if, a failure occurs in any of its routes). Therefore, the overall reliability of a solution with R routes can be computed as $\prod_{r=1}^{R} reliab_r$. This leads to another valuable decision variable for waste collection route planners, especially due to the fact that more than one solution is evaluated in the same manner when applying the described reliability calculation to each elite solution. Furthermore, it allows for a closer risk and sensitivity analysis of the considered solutions, as explained in the following Section.

6. Computational experiments for the stochastic Waste Collection Problem

Similar to the deterministic case, a set of computational experiments have been performed for the WCP under uncertainty, which are described in this Section. Furthermore, the obtained results are discussed and analyzed.

Description of Experiments

Since there is a lack of stochastic benchmark instances with similar characteristics, we use the non-clustered instances of Kim et al. (2006) as reference. The deterministic instances are then transformed into stochastic ones by using random waste levels following a log-normal distribution with expected values equal to the original deterministic value. This probability distribution has been chosen because it is quite flexible and among the most popular ones when modeling non-negative random variables. Other probability distributions, like the normal one, are rarely employed to model non-negative random variables. Nevertheless, our approach could be used with any other probability distribution (e.g., Weibull, gamma, etc.). Note that any probability distribution will allow the easy construction of the deterministic case by putting the variance level $Var[w_i]$ of any container equal to 0, considering that the deterministic values provided by the instances are used as the distribution mean.

We test our approach using low ($Var[w_i] = 0.05$), medium ($Var[w_i] = 0.15$), and high variance levels ($Var[w_i] = 0.25$) concerning the waste level distribution at any container. The number of short simulation runs is set to 500, while a more extensive simulation with 5000 runs is applied only to the elite solutions. Moreover, we propose the inclusion of vehicle safety stocks k to better deal with unexpected demands, as discussed in more detail by Juan et al. (2011b). Instead of considering the complete available vehicle capacity C in the construction of the deterministic solution, a decreased capacity $C^* = C*(1-k)$ is applied. On the one hand, high levels of k will, on average, lead to higher deterministic costs (and increased solution reliabilites), as the considered vehicle capacity during the route construction is reduced. On the other hand, it can be expected that the stochastic route failure costs will decrease. For the following analysis and discussion of results we will therefore consider 6 different safety stock levels k: 0, 0.02, 0.04, 0.06, 0.08, and 0.1. Combined with the three variance levels, this leads to a total of 18 different scenarios for each instance. Tables A.5-A.7 show the deterministic costs (1), the total costs including the expected route failure penalties (2), and the related reliability calculated as described in the previous Section (3) of each tested scenario, where listed results refer to the best obtained solution according to the overall costs. The average calculation time (to complete the VNS procedure and the subsequent extensive simulation for the elite solutions) of all scenarios was 351.92 seconds.

Discussion and Analysis of Results

Figure A.4 shows the expected total costs and reliabilities for the average of all tested instances for each waste variance level/safety capacity factor combination. As can be observed, the highest total costs for each waste variance level is obtained when no safety capacity factor is considered as a result of high expected route failure costs. Furthermore, it can be seen that the lowest total costs over all instances for a low variance level are obtained with a safety capacity factor of 2%. For medium and high waste variance, a safety capacity factor of 4% seems to yield the most promising results concerning total costs. As expected however, the reliability levels (also calculated as an average of all instances) increase for all variance levels as the vehicle safety capacity is increased.

TABLE A.5: Computational results for the stochastic case with a low variance level

Safety Capacity	0%			2%			4%			6%			8%			10%		
Instance	(1) Det Costs	(2) Total Costs	(3) Reliab.	(1) Det Costs	(2) Total Costs	(3) Reliab.	(1) Det Costs	(2) Total Costs	(3) Reliab.	(1) Det Costs	(2) Total Costs	(3) Reliab.	(1) Det Costs	(2) Total Costs	(3) Reliab.	(1) Det Costs	(2) Total Costs	(3) Reliab.
Kim102	158.78	158.78	1	158.55	158.55	1	156.14	156.15	0.9996	157.67	157.67	1	158.56	158.56	1	158.18	158.18	1
Kim277	471.48	512.35	0.2052	462.46	478.32	0.6274	486.71	487.16	0.9864	491.45	491.45	0.9998	493.89	493.89	1	494.7	494.7	1
Kim335	189.19	189.97	0.3142	187.83	188.06	0.8563	187.09	187.11	0.9944	187.65	187.65	1	189.72	189.72	1	189.66	189.66	1
Kim444	80.48	83.23	0.15	84.68	85.1	0.8476	84.52	84.55	0.9886	86.37	86.37	1	88.43	88.43	1	91.06	91.06	1
Kim804	606.48	645.04	0.0323	621.8	627.4	0.7031	624.94	625.03	0.9952	631.61	631.61	1	642.98	642.98	1	638.02	638.02	1
Kim1051	2200.05	2402.75	0	2216.32	2251.19	0.1883	2229.97	2231.21	0.9497	2313.75	2313.75	0.9996	2319.08	2319.08	1	2333.96	2333.96	1
Kim1351	924.17	1014.92	0.0158	954.2	961.61	0.756	978.95	979.31	0.9871	990.21	990.21	1	996.45	996.45	1	1021.53	1021.53	1
Kim1599	1205.23	1296.8	0.002	1209.95	1217.99	0.6641	1242.46	1242.54	0.9934	1270.23	1270.23	1	1276.18	1276.18	1	1283.12	1283.12	1
Kim1932	1144.16	1237.23	0.0008	1149.68	1152.5	0.7698	1182.4	1182.4	1	1197.76	1197.76	1	1209.53	1209.53	1	1243.92	1243.92	1
Kim2100	1599.38	1707.03	0.0002	1679.76	1683.14	0.8555	1640.45	1640.46	0.9996	1655.24	1655.24	1	1678.34	1678.34	1	1699.23	1699.23	1

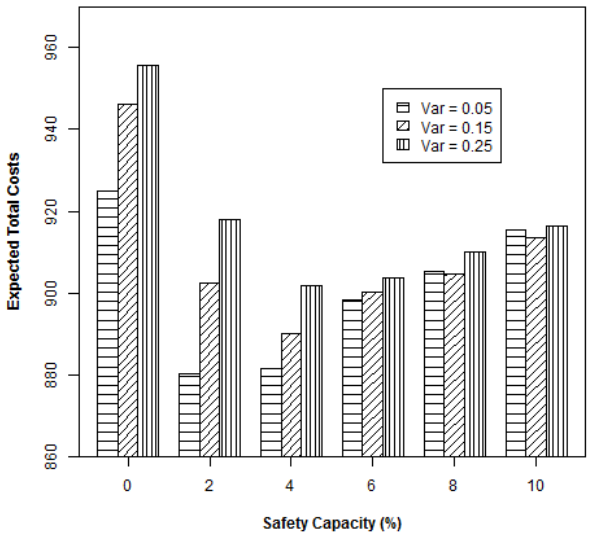
TABLE A.6: Computational results for the stochastic case with a medium variance level

Safety Capacity	0%			2%			4%			6%			8%			10%		
Instance	**(1) Det Costs**	**(2) Total Costs**	**(3) Reliab.**	**(1) Det Costs**	**(2) Total Costs**	**(3) Reliab.**	**(1) Det Costs**	**(2) Total Costs**	**(3) Reliab.**	**(1) Det Costs**	**(2) Total Costs**	**(3) Reliab.**	**(1) Det Costs**	**(2) Total Costs**	**(3) Reliab.**	**(1) Det Costs**	**(2) Total Costs**	**(3) Reliab.**
Kim102	153.27	153.56	0.7681	154.7	154.9	0.9665	158.27	158.37	0.9968	157.67	157.73	0.9982	158.56	158.56	1	158.18	158.18	1
Kim277	462.58	524.87	0.1087	490.06	500.8	0.7097	492.65	498.32	0.8493	496.76	497.48	0.9809	494.29	494.58	0.9922	495.68	495.69	0.9998
Kim335	189.5	190.84	0.3456	187.2	187.71	0.5508	186.21	186.7	0.8249	187.17	187.19	0.9892	189.72	189.72	0.999	189.59	189.59	1
Kim444	80.43	85.53	0.034	84.69	86.8	0.3972	85.16	85.86	0.6948	86.37	86.47	0.9517	87.58	87.59	0.9966	90.84	90.84	0.999
Kim804	607.75	663.12	0.0117	623.64	643.96	0.3229	629.78	633.32	0.8274	630.59	630.79	0.9833	637.11	637.12	0.999	630.04	630.04	1
Kim1051	2201.89	2481.65	0	2215.63	2335.24	0.0026	2251.3	2278.71	0.2778	2309.16	2314.82	0.7669	2319.08	2319.78	0.9712	2333.96	2333.99	0.9988
Kim1351	925.63	1060.93	0.0019	950.02	988.05	0.2058	972.84	982.14	0.6875	1002.1	1002.87	0.9715	996.46	996.51	0.9978	1021.53	1021.53	0.9998
Kim1599	1202.08	1317.4	0.0006	1209.99	1248.56	0.123	1245.18	1251.18	0.6799	1270.23	1271.02	0.9613	1276.18	1276.2	0.9974	1283.12	1283.12	1
Kim1932	1144.16	1248.28	0.0003	1150.33	1171.73	0.1891	1182.4	1184.55	0.8934	1197.76	1197.83	0.9966	1209.53	1209.53	0.9996	1243.92	1243.92	1
Kim2100	1604.81	1735.66	0	1682.09	1707.87	0.1904	1640.79	1642.26	0.901	1655.24	1655.29	0.9962	1678.34	1678.34	1	1687.77	1687.77	1

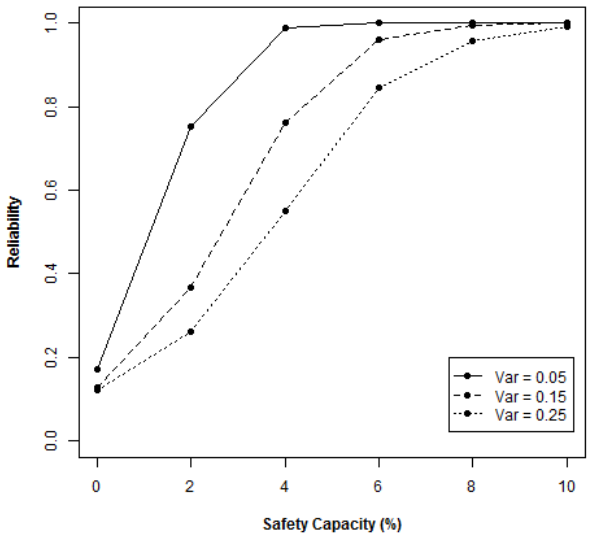
TABLE A.7: Computational results for the stochastic case with a high variance level

Safety Capacity	0%			2%			4%			6%			8%			10%		
Instance	(1) Det Costs	(2) Total Costs	(3) Reliab.	(1) Det Costs	(2) Total Costs	(3) Reliab.	(1) Det Costs	(2) Total Costs	(3) Reliab.	(1) Det Costs	(2) Total Costs	(3) Reliab.	(1) Det Costs	(2) Total Costs	(3) Reliab.	(1) Det Costs	(2) Total Costs	(3) Reliab.
Kim102	158.56	158.57	0.9996	158.27	158.74	0.9842	158.27	158.65	0.9874	156.58	157.02	0.9821	157.21	157.26	0.9984	158.18	158.18	1
Kim277	461.1	537.7	0.0678	487.9	515.37	0.4067	500.22	509.25	0.75	499.31	500.78	0.9588	500.28	502.17	0.9507	498.1	498.7	0.9833
Kim335	189.61	192.76	0.142	186.4	187.86	0.3888	190.09	191.22	0.6926	189.13	189.35	0.8959	189.05	189.08	0.9821	192.09	192.09	0.9982
Kim444	80.43	86.86	0.0154	85.52	87.57	0.3537	85.14	86.83	0.4329	85.92	86.2	0.857	87.88	87.97	0.9524	90.27	90.31	0.9782
Kim804	606.33	670.43	0.0069	624.63	646.11	0.2675	630.17	639.85	0.5774	630.63	632.15	0.8855	643.52	643.93	0.9788	642.68	642.75	0.9968
Kim1051	2204.77	2518.54	0	2215.87	2387.04	0.0001	2247.29	2325.05	0.0238	2311.87	2334.32	0.3647	2319.08	2324.99	0.7783	2333.96	2334.86	0.9627
Kim1351	919.79	1060.65	0.001	950.02	1011.85	0.0728	984.52	1003.82	0.4638	1002.2	1006.84	0.8436	1013.53	1014.49	0.964	1021.53	1021.59	0.9978
Kim1599	1202.08	1329.94	0.0003	1209.99	1268.65	0.0384	1246.46	1263.17	0.3251	1270.81	1275.48	0.7911	1290.52	1291.06	0.9725	1294	1294.1	0.9926
Kim1932	1144.16	1255.11	0.0002	1150.33	1187.7	0.0572	1182.4	1191.29	0.6235	1197.76	1198.99	0.9375	1209.53	1209.59	0.996	1243.55	1243.55	0.9998
Kim2100	1604.81	1746.22	0	1682.09	1728.19	0.0461	1640.79	1647.58	0.6153	1655.24	1656.45	0.9279	1678.34	1678.4	0.9976	1687.77	1687.77	1

It can also be concluded that the inclusion of only a small safety capacity already significantly increases reliability levels (up to around 60% in the most extreme case). In contrast to the stochastic case, safety capacity levels negatively impact the deterministic results as vehicle capacity levels are reduced. This can be clearly seen in Figure A.5, showing the average deterministic costs of all instances and variance levels with different safety stock levels.



(A)



(B)

FIGURE A.4: Two numerical solutions: Expected total costs *(a)* and reliabilities *(b)* over all instances

A more detailed risk analysis is done in Figure A.6, which shows a boxplot of the long simulation outputs for the three most competitive elite solutions of the Kim277 instance. In this specific case the first solution seems to be the most promising one, as it has the lowest mean and the lowest quartiles. However, this is not necessarily always the case. In Table A.8, the mean and standard deviation of the results from the long simulation concerning total costs of the three best solutions of each instance (obtained with a single random-number seed) are listed. From the table it can be concluded that the solution with the lowest mean does not always have the lowest standard deviation (see for example Kim444). Thus, this information can be used by decision-makers to select the solution that he/she prefers according to his/her risk preference. In a similar manner, our solution approach allows the consideration of different risk-aversion levels of decision takers

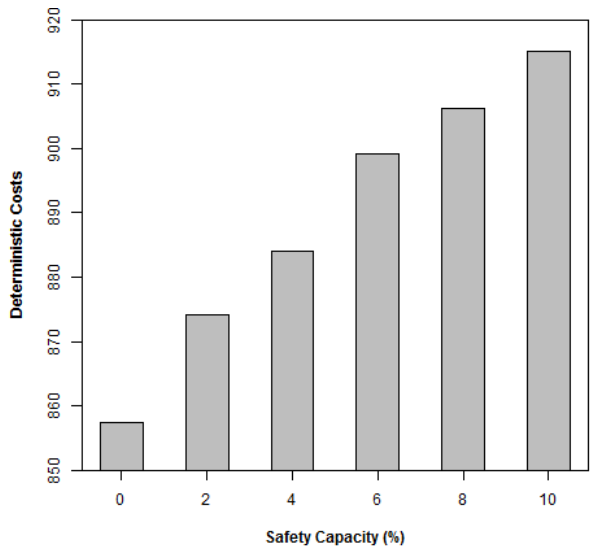


FIGURE A.5: Deterministic costs over all instances and variance levels

by comparing solutions with different safety capacity levels. A more risk-averse route planner will choose to construct routes with higher safety capacity levels, which typically lead to higher routing costs while experiencing lower route failure, and vice versa.

To assess the relationship between reliability levels and associated solution costs, Pearson's product-moment correlation test has been completed by calculating the normalized values of the reliabilities, expected route failure costs, and the total costs of all elite solutions for each instance. Hereby, a 4% vehicle safety capacity and all variance levels have been considered. When comparing reliability levels and total solution costs, only a very weak negative correlation of -0.072 (p-value = 0.3035) can be observed. As could be expected however, the negative correlation between reliability levels and expected route failure costs is more clear, with a Pearson correlation of -0.674 (p-value < 2.2e-16).

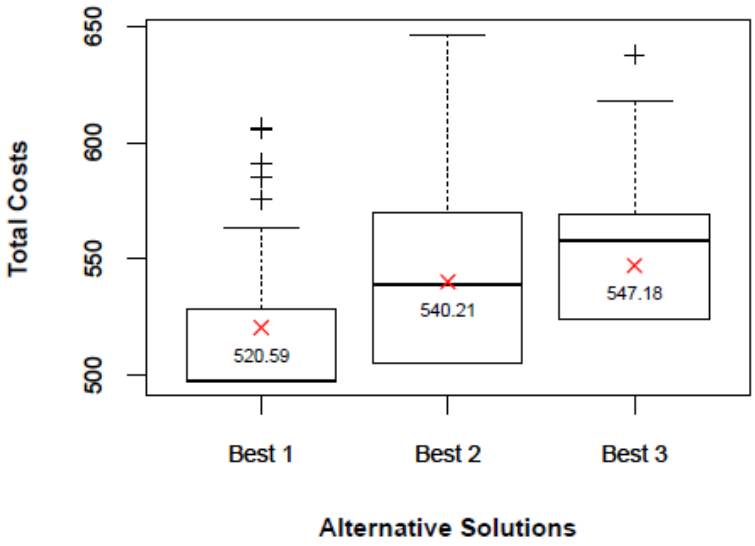


FIGURE A.6: Boxplot of the total costs of each long simulation run of the Kim277 instance for the best three solutions considering a high waste variance level and a 2% safety capacity level

TABLE A.8: Comparison of different elite solutions in terms of the mean and standard deviation of total costs

Elite Solutions	**Best 1**		**Best 2**		**Best 3**	
Instance Name	Mean	St. Dev.	Mean	St. Dev.	Mean	St. Dev.
Kim102	157.05	3.54	157.14	3.38	157.22	3.65
Kim277	498.66	4.53	499.07	4.45	499.12	4.59
Kim335	187.84	1.81	187.96	1.84	188.25	1.85
Kim444	87.79	0.84	87.80	0.79	91.35	0.82
Kim804	633.97	5.93	634.34	5.74	635.00	5.90
Kim1051	2342.85	16.67	2343.58	15.48	2345.62	16.29
Kim1351	1009.88	26.48	1012.78	26.57	1025.50	26.54
Kim1599	1290.02	24.34	1291.67	23.40	1292.07	23.83
Kim1932	1199.85	29.77	1202.21	30.50	1245.03	30.14
Kim2100	1742.47	13.97	1742.81	14.62	1748.34	13.83

7. Conclusions

The Waste Collection Problem (WCP) is becoming increasingly popular due to the growth of urban areas and smart cities around the world. Efficient waste management has relevant benefits for society, such as the reduction of environment pollution, hygiene problems, traffic jams, and direct costs. Formulated as an optimization problem, the WCP can be seen as an extension of the Capacitated Vehicle Routing Problem. Special problem characteristics include the pick-up activities of waste and the inclusion of additional landfill trips. Despite the amount of works in the literature devoted to this problem, most works assume that waste levels are known when designing vehicle routes, which is not the case in real-life applications. In addition to a competitive Variable Neighborhood Search metaheuristic for the deterministic WCP, this paper has presented an efficient approach to solve the WCP under uncertainty by modeling waste levels as random variables following an empirical or theoretical probability distribution. The algorithm is tested using a large-scaled benchmark set for the WCP with several realistic constraints.

The proposed methodology for the WCP with stochastic waste levels is based on a simheuristic algorithm in which a VNS metaheuristic is combined with simulation techniques. Initially, a stochastic problem instance is transformed into a deterministic one by replacing random variables with their means. In the following the metaheuristic explores the search space to find a set of promising solutions, which are then assessed in a stochastic environment by using Monte Carlo simulation. Apart from finding different solutions in only a few minutes (even for stochastic WCP cases with over 2000 nodes solutions are found in under 400 seconds), the results allow a risk analysis considering waste level variances and vehicle safety capacities. A further advantage of our approach is its easiness to be understood and implemented. In addition, no strong assumptions are made related to the probability distribution of the random variables. As the results of our computational experiment show, our algorithm yields competitive solutions in a relatively small amount of time.

A number of possible future research lines stem from this work. The most natural extension would be the inclusion of stochastic travel and/or service times. Especially in urban settings, these variables may experience high uncertainty levels due to the unpredictability of traffic jams, road works, adverse weather, etc. A second research line extends the stochastic problem by considering on-line optimization techniques. In the development of smart cities for example, total waste collection costs could be reduced by using real-time waste level information obtained through volumetric sensors in containers. Another interesting topic would be the introduction of routing externalities (pollution, benefits for society, etc.) in the objective function. Finally, a multi-stage version of our problem (e.g., daily waste collection over a weekly or monthly planning horizon) could be addressed.

Acknowledgments

This work has been partially supported by the Spanish Ministry of Economy and Competitiveness (TRA2013-48180-C3-P, TRA2015-71883-REDT), and FEDER. Likewise, we want to acknowledge the support received by the Department of Universities, Research & Information Society of the Catalan Government (2014-CTP-00001) and the doctoral grant of the UOC.

References

Alshraideh, H. and H. Abu Qdais (2016). "Stochastic modeling and optimization of medical waste collection in Northern Jordan". In: *Journal of Material Cycles and Waste Management*, pp. 1–11.

Baptista, S., R. C. Oliveira, and E. Zúquete (2002). "A period vehicle routing case study". In: *European Journal of Operational Research* 139, pp. 220–229.

Bautista, J., E. Fernández, and J. Pereira (2008). "Solving an urban waste collection problem using ants heuristics". In: *Computers & Operations Research* 35, pp. 3020–3033.

Beliën, J., L. De Boeck, and J. Van Ackere (2014). "Municipal solid waste collection and management problems: a literature review". In: *Transportation Science* 48, pp. 78–102.

Beltrami, E. J. and L. D. Bodin (1974). "Networks and vehicle routing for municipal waste collection". In: *Networks* 4, pp. 65–94.

Benjamin, A. M. and J. E. Beasley (2010). "Metaheuristics for the waste collection vehicle routing problem with time windows, driver rest period and multiple disposal facilities". In: *Computers & Operations Research* Vol. 37, pp. 2270–2280.

Bianchi, L., M. Birattari, M. Chiarandini, M. Manfrin, M. Mastrolilli, L. Paquete, O. Rossi-Doria, and T. Schiavinotto (2006). "Hybrid metaheuristics for the vehicle routing problem with stochastic demands". In: *Journal of Mathematical Modelling and Algorithms* 5, pp. 91–110.

Bianchi, L., D. Marco, L. M. Gambardella, and W. J. Gutjahr (2009). "A survey on metaheuristics for stochastic combinatorial optimization". In: *Natural Computing* 8, pp. 239–287.

Buhrkal, K., A. Larsen, and S. Ropke (2012). "The waste collection vehicle routing problem with time windows in a city logistics context". In: *Procedia - Social and Behavioral Sciences* 39, pp. 241–254.

Caceres, J., P. Arias, D. Guimarans, D. Riera, and A. A. Juan (2014). "Rich vehicle routing problem: a survey". In: *ACM Computing Surveys* 47.2. ISSN: 0360-0300.

Christofides, N. and J. E. Beasley (1984). "The period routing problem". In: *Networks* 14, pp. 237–256.

Clarke, G. and J. Wright (1964). "Scheduling of vehicles from a central depot to a number of delivering points". In: *Operations Research* 12, pp. 568–581.

Corberán, Á. and G. Laporte (2014). *Arc routing: problems, methods, and applications*. SIAM Monographs on Discrete Mathematics and Applications.

Faccio, M., A. Persona, and G. Zanin (2011). "Waste collection multi objective model with real time traceability data". In: *Waste Management* 31, pp. 2391–2405.

Faulin, J., M. Gilibert, A. A. Juan, X. Vilajosana, and R. Ruiz (2008). "A simulation-based algorithm for the capacitated vehicle routing problem". In: *Proceedings of the 2008 Winter Simulation Conference*. Miami, US, pp. 2708–2716.

Fikar, C., A. A. Juan, E. Martinez, and P. Hirsch (2016). "A discrete-event driven metaheuristic for dynamic home service routing with synchronised trip sharing". In: *European Journal of Industrial Engineering* 10, pp. 323–340.

Gendreau, M., G. Laporte, and R. Séguin (1996). "Stochastic vehicle routing with modified savings algorithms". In: *European Journal of Operational Research* 88, pp. 3–12.

Gendreau, M., O. Jabali, and W. Rei (2014). "Stochastic vehicle routing problems". In: *Vehicle Routing: Problems, Methods, and Applications*. Ed. by P. Toth and D. Vigo. 2nd. SIAM Monographs on Discrete Mathematics and Applications, pp. 213–239.

Ghiani, G., F. Guerriero, G. Improta, and R. Musmanno (2005). "Waste collection in Southern Italy: solution of a real-life arc routing problem". In: *International Transactions in Operational Research* 12, pp. 135–144.

Ghiani, G., D. Laganá, E. Manni, R. Musmanno, and D. Vigo (2014a). "Operations research in solid waste management: a survey of strategic and tactical issues". In: *Computers & Operations Research* 44, pp. 22–32.

Ghiani, G., C. Mourão, L. Pinto, and D. Vigo (2014b). "Routing in waste collection applications". In: *Arc routing: problems, methods, and applications*. Ed. by A. Corberán and G. Laporte. Philadelphia: SIAM Monographs on Discrete Mathematics and Applications, pp. 351–370.

Gonzalez, S., D. Riera, A. A. Juan, M. G. Elizondo, and P. Fonseca (2012). "Proceedings of the 2012 Winter Simulation Conference". In: *SIM-RANDSHARP: a hybrid algorithm for solving the arc routing problem with stochastic demands*. Ed. by C. Laroque, J. Himmelspach, R. Pasupathy, O. Rose, and A. M. Uhrmacher, pp. 1–12.

Gonzalez, S., A. A. Juan, D. Riera, M. Elizondo, and J. Ramos (2016). "A simheuristic algorithm for solving the arc routing problem with stochastic demands". In: *Journal of Simulation*.

Gruler, A., C. Fikar, A. A. Juan, P. Hirsch, and C. Contreras-Bolton (2015a). "A simheuristic for the waste collection problem with stochastic demands in smart cities". In: *Proceedings of the 16. ASIM dedicated conference on simulation*. Dortmund, Germany, pp. 49–58.

Han, H. and E. Ponce-Cueto (2015). "Waste collection vehicle routing problem: literature review". In: *Traffic & Transportation* 27, pp. 345–358.

Hansen, P., N. Mladenovic, and J. A. Moreno (2010b). "Variable neighbourhood search: methods and applications". In: *Annals of Operations Research* 175.1, pp. 367–407.

Hemmelmayr, V., Karl F. Doerner, R. F. Hartl, and S. Rath (2013). "A heuristic solution method for node routing based solid waste collection problems". In: *Journal of Heuristics* 19, pp. 129–156.

Hemmelmayr, V., K. F. Doerner, R. F. Hartl, and D. Vigo (2014). "Models and algorithms for the integrated planning of bin allocation and vehicle routing in solid waste management". In: *Transportation Science* 48, pp. 103–120.

Ismail, Z. and I. Irhamah (2008). "Solving the vehicle routing problem with stochastic demands via hybrid genetic algorithm-tabu search". In: *Journal of Mathematics and Statistics* 4, pp. 161–167.

Ismail, Z. and S.L. Loh (2009). "Ant colony optimization for solving solid waste collection scheduling problems". In: *Journal of Mathematics and Statistics* 5, pp. 199–205.

Johansson, O. M. (2006). "The effect of dynamic scheduling and routing in a solid waste management system". In: *Waste Management* 26, pp. 875–885.

Juan, A. A., J. Faulin, S. Grasman, D. Riera, J. Marull, and C. Mendez (2011b). "Using safety stocks and simulation to solve the vehicle routing problem with stochastic demands". In: *Transportation Research Part C: Emerging Technologies* 19.5, pp. 751–765.

Juan, A. A., J. Faulin, A. Ferrer, H. Lourenço, and B. Barrios (2013a). "MIRHA: multi-start biased randomization of heuristics with adaptive local search for solving non-smooth routing problems". In: *TOP* 21, pp. 109–132.

Juan, A. A., J. Faulin, J. Jorba, J. Caceres, and J. M. Marques (2013b). "Using parallel & distributed computing for real-time solving of vehicle routing problems with stochastic demands". In: *Annals of Operations Research* 207.1, pp. 43–65.

Juan, A. A., B. Barrios, E. Vallada, D. Riera, and J. Jorba (2014a). "A simheuristic algorithm for solving the permutation flow shop problem with stochastic processing times". In: *Simulation Modelling Practice and Theory* 46, pp. 101–117.

Juan, A. A., S. E. Grasman, J. Caceres-Cruz, and T. Bektaş (2014b). "A simheuristic algorithm for the single-period stochastic inventory-routing problem with stock-outs". In: *Simulation Modelling Practice and Theory* 46, pp. 40–52.

Juan, A. A., J. Faulin, S. Grasman, M. Rabe, and G. Figueira (2015a). "A review of simheuristics: extending metaheuristics to deal with stochastic optimization problems". In: *Operations Research Perspectives* 2, pp. 62–72.

Kim, B., S. Kim, and S. Sahoo (2006). "Waste collection vehicle routing problem with time windows". In: *Computers & Operations Research* Vol. 33, pp. 3624–3642.

Malakahmad, A., P. Md. Bakri, M. R. Md. Mokhtar, and N. Khalil (2014). "Solid waste collection routes optimization via GIS techniques in Ipoh City, Malaysia". In: *Procedia Engineering* Vol. 77, pp. 20–27.

Markov, I., S. Varone, and M. Bierlaire (2015). *The waste collection VRP with intermediate facilities, a heterogeneous fixed fleet and a flexible assignment of origin and destination depot*. Tech. rep. CH: Transport and Mobility Laboratory, Ecole Polytechnique Federale de Lausanne.

— (2016). "Integrating a heterogeneous fixed fleet and a flexible assignment of destination depots in the waste collection VRP with intermediate facilities". In: *Transportation Research Part B: Methodological* 84, pp. 256–273.

Montgomery, D. C. (2008). *Design and analysis of experiments*. 8th ed. John Wiley & Sons.

Nations, United (2015). *World population prospects: the 2015 revision, key findings and advance tables*. Tech. rep. Department of Economic and Social Affairs.

Neirotti, P., A. De Marco, A. C. Cagliano, G. Mangano, and F. Scorrano (2014). "Current trends in smart city initiatives: some stylised facts". In: *Cities* 38, pp. 25–36.

Nolz, P., N. Absi, and D. Feillet (2014). "A stochastic inventory routing problem for infectious medical waste collection". In: *Networks* 63, pp. 82–95.

Nuortio, T., J. Kytöjoki, H. Niska, and O. Bräysy (2006). "Improved route planning and scheduling of waste collection and transport". In: *Expert Systems with Applications* 30, pp. 223–232.

Ombuki-Berman, B. M., A. Runka, and F. T. Hanshar (2007). "Waste collection vehicle routing problem with time windows using multi-objective genetic algorithms". In: *Proceedings of the third IASTED International Conference on Computational Intelligence*. Anaheim, US, pp. 91–97.

Pillac, V., M. Gendreau, C. Gueret, and A. Medaglia (2013). "A review of dynamic vehicle routing problems". In: *European Journal of Operational Research* 225, pp. 1–11. ISSN: 03772217.

Ramos, T. R. P., M. I. Gomes, and A. P. B.arbosa-Póvoa (2014). "Economic and environmental concerns in planning recyclable waste collection systems". In: *Transportation Research Part E: Logistics and Transportation Review* 62, pp. 34–54.

Ritzinger, U., J. Puchinger, and R. F. Hartl (2016). "A survey on dynamic and stochastic vehicle routing problems". In: *International Journal of Production Research* 54, pp. 215–231.

Sahoo, S., S. Kim, B.-I. Kim, B. Kraas, and A. Popov. Jr. (2005). "Routing optimization for waste management". In: *Interfaces* 35, pp. 24–36.

Solomon, M. M. (1987). "Algorithms for the vehicle routing and scheduling problems with time windows". In: *Operations Research* Vol. 35, pp. 254–265.

Son, L. H. (2014). "Optimizing municipal solid waste collection using chaotic particle swarm optimization in GIS based environments: A case study at Danang city, Vietnam". In: *Expert Systems with Applications* 41, pp. 8062–8074.

Tavares, G., Z. Zsigraiova, V. Semiao, and M. Carvalho (2009). "Optimisation of MSW collection routes for minimum fuel consumption using 3D GIS modelling". In: *Waste Management* 29, pp. 1176–1185.

Teixeira, J., A. P. Antunes, and J. P. de Sousa (2004). "Recyclable waste collection planning-a case study". In: *European Journal of Operational Research* 158, pp. 543–554.

The World Bank (2012). *What a waste – A global review of solid waste management*. Tech. rep. URL: http://siteresources.worldbank.org/INTURBANDEVELOPMENT/Resources/336387-1334852610766/What_a_Waste2012_Final.pdf.

Toth, P. and D. Vigo (2014). *Vehicle routing - problems, methods and applications*. Ed. by P. Toth and D. Vigo. 2nd. ISBN: 978-1-61197-358-7.

Wang, F.S. (2001). "Deterministic and stochastic simulations for solid waste collection systems – A SWIM approach". In: *Environmental Modeling & Assessment* 6, pp. 249–260.

Yang, W. H., K. Mathur, and R. H. Ballou (2000). "Stochastic vehicle routing problem with restocking". In: *Transportation Science* 34, pp. 99–112.

Yeomans, J. S. (2007). "Solid waste planning under uncertainty using evolutionary simulation-optimization". In: *Socio-Economic Planning Sciences* 41, pp. 38–60.

Appendix B

Journal papers under review in ISI JCR

B.1 A VNS-based simheuristic methodology for the stochastic portfolio optimization problem

Laura Calvet[1], Renatas Kizys[2], Angel A. Juan[1], Jana Doering[3]

1. Computer Science Department, Open University of Catalonia – IN3, Barcelona, Spain
e-mail: {lcalvetl, ajuanp}@uoc.edu
2. Subject Group of Economics and Finance, University of Portsmouth, UK
e-mail: renatas.kizys@port.ac.uk
3. Economics and Business Department, Open University of Catalonia – IN3, Barcelona, Spain
e-mail: jdoering@uoc.edu

Abstract

The goal of the portfolio optimization problem is to minimise risk for an expected portfolio return by determining the proportions of included assets. As the pool of assets increases and additional constraints are considered, the problem becomes NP-hard. Thus, metaheuristics are commonly employed for solving large instances of rich versions. However, metaheuristics fail to account for stochastic returns and covariances, rendering them unrealistic in the presence of heightened uncertainty in financial markets. This paper aims at closing this gap by proposing a simulation-optimization approach, specifically a simheuristic algorithm that integrates a variable neighbourhood search metaheuristic with Monte Carlo simulation, to deal with returns and covariances modelled as random variables. Computational experiments performed on a well-established benchmark instance illustrate the use of our methodology and analyse how the solutions change in response to varying the degree of randomness, minimum required return, and probability of obtaining at least the specified return.

Keywords: constrained portfolio optimization, metaheuristics, efficiency indices, financial assets, iterated local search, biased randomization.

1. Introduction

Investments play an essential role in our society through wealth creation, sustainable economic growth and ultimately improvements in welfare standards. They provide companies with the necessary funds to transform ideas and resources into profitable projects, social benefits and jobs. Most of the related questions in financial economics can be formulated as combinatorial optimization problems (COPs).

Optimization methods may be classified into exact methods and heuristics/metaheuristics (Talbi, 2009). The first group includes procedures that guarantee the optimality of a solution. However, exact methods may require strict assumptions or simplifying formulations to render it solvable, thus reducing the informative value for real-life operational applications. Furthermore, when they are used to solve large instances of NP-hard optimization problems, they require enormous computing times. Within the second group, heuristics are experience-based procedures, which usually provide reasonably good solutions in very short computing times (a few seconds

or even less). By contrast, metaheuristics (Boussaïd et al., 2013) are general solving procedures, which are able to provide near-optimal solutions for a broad range of optimization problems in reasonable computing times (typically in the order of a few minutes).

Founded by Harry Markowitz (Markowitz, 1952) and a milestone in modern portfolio theory, the portfolio optimization problem (POP) defines the investment decision as the strategy of: *(i)* selecting financial assets; and *(ii)* determining the optimal weights allocated to those assets that results in a desired portfolio return and an associated minimum level of risk. This is implemented through a quadratic objective function that aggregates the weighted covariances of the constituent asset returns, which is then minimised subject to a desired rate of return. It is noteworthy that other risk measures, such as value-at-risk or Sharpe ratio have been employed too. In its most basic definition the POP is constrained in that portfolio weights must add up to one and take on non-negative values, prohibiting short-selling. This basic version of the problem can be solved through exact methods and, in fact, these methods have been predominant in the POP literature (Mansini et al., 2014; Sawik, 2012a). However, metaheuristics are increasingly employed to deal with more realistic and complex versions of the problem (Adebiyi and Ayo, 2015), in which additional constraints are considered. In particular, pre-assignment, quantity, and cardinality constraints have received overwhelming attention in extant literature (Chang et al., 2000; Soleimani et al., 2009). Pre-assignment constraints allow the preselection of some assets, irrespective of their risk-return characteristics. Quantity constraints confine the weight allocated to an asset in the portfolio within a floor and ceiling constraint, thus simultaneously limiting the exposure to specific assets and ruling out investments in negligible quantities –a practice that may be prohibitively costly, since the transaction costs may reduce or erase the benefit. While recognising that the quantity constraints arise as a result of the investor's discretionary decisions, these constraints have received growing interest. A recent example is Babaei et al. (2015), which is further supported by Kolm et al. (2014). The latter authors argue that the inclusion of quantity constraints can lead to an improved performance, can help contain portfolio volatility, and can decrease downside risk and shortfall probability. Behr et al. (2013) further assert that tightening these constraints helps ensure that portfolio weights are not driven by the sampling error that stems from parameter estimates based on historical data. Finally, cardinality constraints determine a lower and upper bound for the number of assets included in the portfolio. Diversification is to some extent ensured through the allocation of resources to a minimum number of imperfectly correlated assets, while the upper bound is dictated by the trade-off between diversification and incurred costs. After a certain threshold the marginal benefits of portfolio diversification decrease (Maringer, 2005), while portfolios with a large number of assets are more costly in terms of complexity, managerial effort, and the ensuing increased transaction costs. All in all, these additional constraints make the problem NP-hard (Bienstock, 1996), thus requiring the use of metaheuristics.

Contrary to the well-established real-life constraints, the growing body of literature assumes constant rates of returns and covariances. This empirically unsupported assumption poses a key limitation when real-life approaches are sought, and the main contribution of this paper is to address this limitation. Indeed, since asset returns are random variables that obey certain probability density functions, and future returns are unpredictable, the minimum desired rate of return may not be attained with certainty. More concretely, we relax the above simplifying assumptions and consider rates of returns and covariances as random variables. The resulting problem is known as the stochastic POP (SPOP). Figure B.1 shows two resulting portfolios with a different required return for Hang Seng Stock Market data with a medium level of stochasticity for covariances (in this illustrative example, returns are maintained constant). As expected, the higher the required return, the higher the expected risk and the lower the number of assets selected (only those with a relatively high expected return are considered).

The relation between the required return and the expected risk (i.e., the constrained efficient frontier or CEF) is analysed in more detail in Figure B.2 for four different levels of stochasticity (zero, low, medium, and high). The circles on the vertical axis indicate the required returns of the two solutions analysed in the previous figure, while crossings identify the expected risk of the first solution for each level of stochasticity.

In this paper, we propose a simheuristic algorithm to solve the SPOP. As described in Juan et al. (2015a), simheuristic algorithms integrate metaheuristics with simulation techniques in order to deal with the random nature of stochastic COPs. In particular, we use an extension of the variable neighbourhood search (VNS) metaheuristic (Hansen et al., 2010a) that integrates Monte Carlo simulation (MCS) techniques. In short, while the metaheuristic generates promising portfolios for

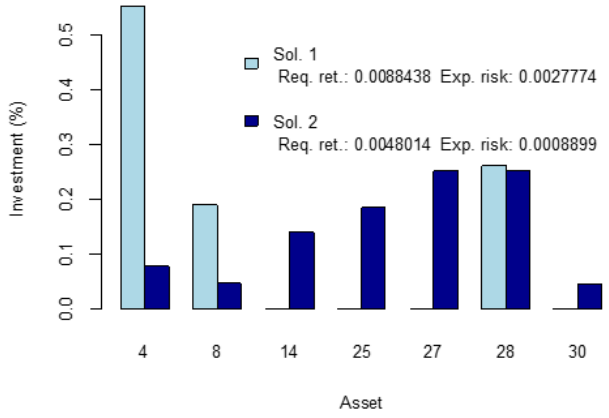


FIGURE B.1: Representation of two solutions for the SPOP.

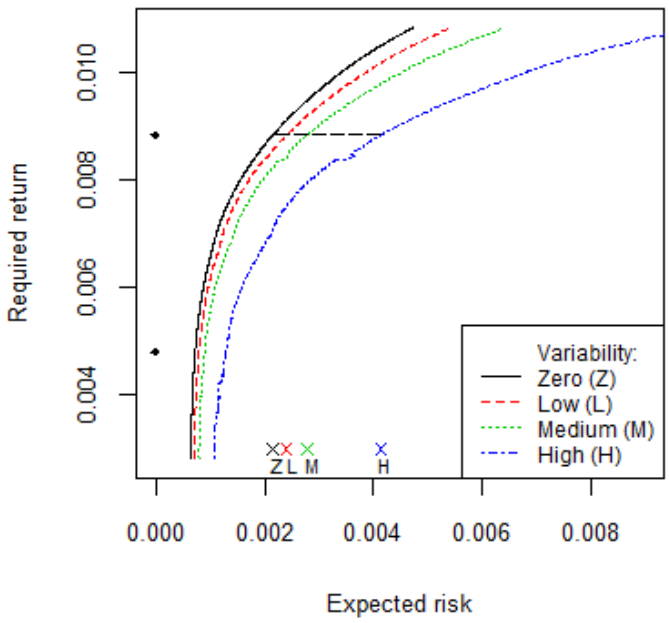


FIGURE B.2: CEFs of Hang Seng Stock Market (Hong Kong) data.

a deterministic version of the problem -the one obtained when expected values are considered-, simulation techniques are applied to: *(i)* estimate the expected risk of these portfolios under uncertainty conditions; *(ii)* complete a risk analysis on each portfolio; and *(iii)* provide feedback to the metaheuristic in order to better guide the searching process. All in all, the main contributions of this paper are: *(i)* to derive a mathematical formulation for the stochastic POP; *(ii)* to develop an efficient solving methodology for it; and *(iii)* to illustrate its use by solving an adapted benchmark instance.

The remainder of this paper is organised as follows. The next section presents a literature review. Afterwards, the descriptions and the mathematical models for the POP and the SPOP are provided. Then, a solving methodology for the SPOP is proposed. The following sections explain the computational experiments carried out and analyse the results. Finally, the last section gathers the main conclusions of this work.

2. Literature review

Implicit in the mean-variance efficient frontier (MVEF) proposed by Markowitz (1952) are the assumptions that: *(i)* first and second moments of a return distribution do not vary over time; and *(ii)* current and past asset price changes can be used to predict future movements in financial asset prices. It is worth noting that the first assumption has received very little empirical support in the related literature, whereas the second rules out the efficient market hypothesis in the semi-strong efficiency form. Even so, the MVEF has received an overwhelming interest from academics and practitioners for more than six decades and has spawn a large number of theoretical extensions and practical applications (Kolm et al., 2014). In fact, and despite the aforementioned shortcomings, there has been little attempt to reconcile the MVEF with the random and uncertain nature of financial asset returns. Early research addresses uncertainty in the parameters, i.e., risk, return, and the covariance matrix of asset returns, through robust model formulation of the mean-variance optimization (Bertsimas and Thiele, 2006). However, these formulations present a deterministic uncertainty model in that it cannot adopt different uncertainty sets (Dangi, 2013).

Huang (2007) addresses this issue and defines returns not as random, but as uncertain and exposed to subjective imprecisions. The author thus combines the MVEF with uncertainty theory and bases returns mainly on expert opinions rather than historical data. A genetic algorithm (GA) is used in order to optimise the resulting mean-variance and mean-semivariance models. However, Qin (2015) argues that a distinction between different securities is to be made and addresses the real-world complexity of financial markets by solving a hybrid POP that distinguishes between random and uncertain returns. If there is ample historical information available on a security, then the security return is considered a random variable. If however there exists a lack of historical information on a security –which applies to newly listed securities on a stock exchange– then the return is treated as fuzzy or uncertain. In the latter case, to estimate the first and second moments of asset returns experts' opinions are sought. In the presence of assets with random and uncertain returns, the author derives an optimal solution of the POP that comprises assets from Shanghai Stock Exchange. Although the aforementioned work is an interesting and significant extension of the MVEF, it ignores several fundamental issues. First, in addition to the presence of random and uncertain asset returns, the investor might be less concerned with keeping a hypothetical expected rate of return on or above a certain threshold. Indeed, it is highly unlikely that all the individual assets will take on average returns at the end of the investment period. It would be more natural and practical if the investor instead maximises the probability that the actual return on a portfolio investment will be on or above a certain threshold. Second, in real life investors face additional constraints, such as cardinality, quantity and pre-assignment constraint. These issues are echoed in Nazemi et al. (2015). They formulate and solve several POPs under the assumption that returns behave as normal, rectangular and trapezoidal uncertain variables. This assumption allows reducing the POPs to linear programming (LP) problems. A neural network model is then designed to solve dynamic version of the POPs. An alternative measure of return uncertainty is introduced by Ning et al. (2015). In the standard MVEF, they consider a POP with triangular entropy as an additional constraint that controls the level of uncertainty in the POP, whilst leaving unresolved the above mentioned issues. The first issue is partially alleviated by Huang (2007), who supports the argument of maximizing actual return on portfolio as opposed to maximizing expected value of portfolio returns. Consequently, the author designs a hybrid intelligent algorithm to solve a

SPOP, which seeks to maximise the actual portfolio return with certain probability under the constraint that the ratio of the expected portfolio return to the variance of portfolio returns is greater than a pre-set tolerance level. Whilst the article considers stochastic asset returns, it continues to treat the second moments as constant. Moreover, the author's assumption that the investors face no further constraints in the POP still remains an interesting area of opportunity. Along similar lines, Liu et al. (2012a) treat asset return as a fuzzy random variable and formulates several POPs in hybrid uncertain decision systems. The first optimization problem consists in minimizing the variance of the portfolio return subject to a chance constraint. The chance constraint specifies a desired rate of return that can be attained under a prescribed chance level. The second problem caters to tail risk aversion. Specifically, if only the tail of a chance distribution matters for a highly risk averse investor, then she will maximise the chance that fuzzy random return on a portfolio is at least as high as a desired rate of return under the maximum acceptable level of risk. To solve the above optimization problems, these authors employ a combination of MCS and a particle swarm optimization (PSO) algorithm. Concretely, the simulation method is used to simulate probability density functions, whereas the PSO algorithm is utilised to solve stochastic programming problems. They further compare the computational results obtained with the chance-variance based SPOP and with a conventional mean-variance based POP and identify notable differences in optimal portfolio weights across the two settings. While they also continue to treat the second moments as constant, the authors account for uncertainty in the constraints (chance or variance constraint depending on the risk awareness of the investor) by further employing MCS in the simulation of the probability density functions of the constraints. Zhang et al. (2012) further explore fuzzy modelling of returns and combine it with the mean-semivariance-entropy approach in a multi-period setting, in which they explicitly consider four dimensions to the POP, namely the level of risk and return, transaction costs and the degree of diversification of the portfolio. The resulting model is solved using a hybrid algorithm that combines GA and simulated annealing (SA). The entropy measure for diversification is further employed in a mean-variance entropy approach by Chen (2014b). Unlike the former authors, the latter author further employs uncertainty theory in describing investment risk as uncertain portfolio variance and solves the resulting model using an artificial bee colony algorithm (ABC). The shortcomings of previous research are diminished by modelling second moments as uncertain variables and considering further constraints, namely transaction costs. Recognizing that not only returns are subject to uncertainty, Nguyen et al. (2015) develops a model, in which the uncertainty of the fuzzy portfolio returns is minimised, while the Sharpe ratio (as simultaneous measure of risk and profitability) is initiated in a fuzzy context and subsequently maximised. The resulting model is solved twice, employing a fuzzy approach and a GA, and both provide superior solutions to the traditional mean-variance optimization based on the workings by Markowitz (1952).

Based on previous work recognizing the importance of stochastic modelling of returns, this paper takes a further important step in acknowledging stochasticity not only in the stock returns, but also in the objective function by considering the covariances random variables. Furthermore, it combines a novel simheuristic approach with rich constraints that have not previously been considered in stochastic POPs, namely pre-selection, quantity, and cardinality constraints.

3. Problem definition

This section provides the descriptions of the POP and SPOP introducing the notation employed and the mathematical formulations, which are based on Di Tollo and Roli (2008).

Notation and description

In the POP, there is a set $A = \{a_1, a_2, \ldots, a_n\}$ of n assets. Each asset a_i ($\forall i \in \{1, 2, \ldots, n\}$) is characterised by an expected return r_i. The covariance between two assets a_i and a_j ($\forall i, j \in \{1, 2, \ldots, n\}$) is denoted by σ_{ij}. A solution for this problem consists of a vector $X = (x_1, x_2, \ldots, x_n)$, where each element $x_i (0 \leq x_i \leq 1)$ represents the weight or fraction of the investment allocated to the asset a_i. The basic aim is to minimise the portfolio risk while obtaining an expected return greater than an investor-given threshold R. To adjust for real-life settings, the rich version of the POP considers pre-selection, quantity, and cardinality constraints. Pre-selection constraints dictate whether an asset a_i has been pre-selected by the investor and is to be included in the solution (i.e., $x_i > 0$) by means of the parameter p_i: $p_i = 1$ if a_i is pre-selected, and $p_i = 0$ otherwise. Quantity

constraints specify a lower and an upper bounds for the weights x_i, ε_i and δ_i ($0 \leq \varepsilon_i \leq \delta_i \leq 1$), respectively. Finally, the cardinality constraint defines the minimum and maximum number of assets included in the portfolio, k_{min} and k_{max} ($1 \leq k_{min} \leq k_{max} \leq n$).

The difference between the POP and the SPOP considered in this paper lies in the modelling of asset returns and covariances. While they are represented by expected values in the first case, the second considers realistic stochastic uncertainty and thus treats these as random variables. This results in a modified return constraint where a return no lower than R must be achieved with a probability of, at least, P_0.

Mathematical model

The POP can mathematically be defined as follows:

$$\min f(x) = \sum_{i=1}^{n} \sum_{j=1}^{n} \sigma_{ij} x_i x_j \tag{B.1}$$

subject to:

$$\sum_{i=1}^{n} r_i x_i \geq R \tag{B.2}$$

$$\sum_{i=1}^{n} x_i = 1 \tag{B.3}$$

$$\varepsilon_i z_i \leq x_i \leq \delta_i z_i, \forall i \in \{1, 2, \ldots, n\} \tag{B.4}$$

$$0 \leq \varepsilon_i \leq \delta_i \leq 1, \forall i \in \{1, 2, \ldots, n\} \tag{B.5}$$

$$z_i \leq M x_i, \forall i \in \{1, 2, \ldots, n\} \tag{B.6}$$

$$p_i \leq z_i, \forall i \in \{1, 2, \ldots, n\} \tag{B.7}$$

$$k_{min} \leq \sum_{i=1}^{n} z_i \leq k_{max} \tag{B.8}$$

$$z_i \in \{0, 1\}, \forall i \in \{1, 2, \ldots, n\} \tag{B.9}$$

As pointed out before, the objective function in Equation (1) quantifies the riskiness of the investment and is to be minimised. Equation (2) guarantees that the expected return of the investment does not fall below the threshold R. Equation (3) restrains portfolio investment to existing and pre-defined resources. An auxiliary variable is introduced to indicate whether the asset a_i is included in the solution ($z_i = 1$; $z_i = 0$ otherwise). For all assets a_i, Equation (4) sets lower and upper bounds (ε_i and δ_i, respectively) for x_i in case the asset is selected (i.e., $z_i = 1$). The two bounds are themselves bound by zero and one inclusive (Equation 5). In Equation (6), M is a very large positive value such that $Mx_i \geq 1$ for all i if $x_i > 0$. Equation (7) defines the pre-assignment constraint, where z_i depends on the parameter p_i. If the asset a_i is pre-selected (i.e., $p_i = 1$), it must be included in the solution (i.e., $z_i = 1$) irrespective of its risk-return characteristics. Equation (8) describes the cardinality constraint. Finally, Equation (9) defines z_i as a binary variable.

The mathematical formulation for the SPOP requires two modifications:

- Covariances (C_{ij}) in the objective function are considered to be random variables following a given probability distribution (e.g., the one that best fits the historical data available):

$$f(x) = \sum_{i=1}^{n} \sum_{j=1}^{n} C_{ij} x_i x_j \geq R \tag{B.10}$$

- Equation (2) is replaced by the following probabilistic constraint:

$$P(\sum_{i=1}^{n} R_i x_i \geq R) \geq P_0 \tag{B.11}$$

where R_i refers to the asset return modeled as a random variable. It ensures that the portfolio return will be no lower than the threshold R with a probability of, at least, P_0.

4. Proposed methodology for the SPOP

In this section, we propose a simheuristic algorithm to address the SPOP previously introduced. Generally, this class of algorithms relies on two facts: *(i)* a stochastic COP can be considered a generalization of the deterministic COP, meaning that the latter can be interpreted as a special case of the former when variances are zero; and *(ii)* although stochastic COPs have not been extensively studied (Bianchi et al., 2009), there is usually a vast literature on the associated deterministic COPs. Thus, the approach suggests selecting an efficient algorithm for the deterministic version of a problem and extending it in a natural way by hybridizing it with simulation techniques. This section introduces the proposed simheuristic as well as relevant elements of its implementation in order to allow reproducibility.

VNS metaheuristic and implementation components

As a base framework we employ the VNS metaheuristic, which was first proposed by Mladenović and Hansen (1997). Besides being a popular metaheuristic in combinatorial as well as global optimization, it has been used in a wide range of research fields such as scheduling, vehicle routing, telecommunications, biology, and artificial intelligence. For extensive reviews on applications, the reader is referred to Moreno-Vega and Melián (2008), Hansen et al. (2008a), and Hansen et al. (2008b). In essence, the VNS metaheuristic proposes systematic changes of neighbourhood to find a local minimum by intensifying the search, and to escape from the associated valley by diversifying. It relies on three facts (Hansen et al., 2010a): *(i)* a local minimum with respect to one neighbourhood structure is not necessarily so for another; *(ii)* a global minimum is a local minimum with respect to all possible neighbourhood structures; and *(iii)* for many problems, local minima with respect to one or several neighbourhoods are relatively close to each other. Pseudo-code 1 shows a simple version of the VNS used in this work. Its inputs are the problem instance to solve, the number of neighbourhoods considered (K) and the maximum computational time (T). Frequently, K is set to two or three, and the neighbourhoods are nested. First, the variable t for measuring the time is initialised at zero. Afterwards, an initial solution is obtained and stored in *currentSol*. An outer loop sets the current neighbourhood to the first one and controls that the time-based constraint is satisfied. Inside, another loop builds and tests new solutions. Within this loop, the current solution is initially shaken (or perturbed), generating a solution from the kth neighbourhood of *currentSol*. The resulting solution is stored in *newSol*, which is then improved by means of a local search. If there is an improvement (i.e., *newSol* is preferred over *currentSol*), *newSol* is copied into *currentSol* and the current neighbourhood is set to the first. This constitutes a descendent phase aimed to find a local minimum. Otherwise, the next neighbourhood is analysed (i.e., k is set to $k + 1$). The inner loop is executed until the last neighbourhood is explored (i.e., $k = K$). Finally, *currentSol* is returned.

Our implementation of the VNS metaheuristic includes biased randomization techniques (Juan et al., 2011a), referring to the introduction of randomization in classical deterministic heuristics in order to obtain a number of solutions. Heuristics include at least one step in which one element is to be selected from a list sorted according to a specific criterion. Typically, choosing the first is expected to be the optimal option to construct a high-quality solution. For instance, in the permutation flow-shop problem (Fernandez-Viagas and Framinan, 2015c), which determines a sequence of jobs with the aim of minimizing a time-related measure, the classical NEH heuristic creates a list of jobs that are ordered from more to less time-consuming, iteratively extracts the first, and then chooses the best allocation in the solution. While this is an intuitively logical procedure, it does not necessarily lead to the best solution. In Juan et al. (2014f), the authors propose assigning a probability to each element in the sorted list according to the measure of reference (i.e., the higher in the list an item is ranked, the higher the probability of choosing the

Algorithm 1 Basic structure of the VNS metaheuristic.

```
    VNS(instance, K, T)
 1: t ← 0
 2: initSol ← initialSolution(instance)
 3: currentSol ← initSol
 4: while {t < T} do
 5:     k ← 1
 6:     while {k ≤ K} do
 7:         newSol ← shake(currentSol,k)
 8:         newSol ← localSearch(newSol)
 9:         if {newSol > currentSol} then
10:             currentSol ← newSol
11:             k ← 1
12:         elsek ← k+1
13:         end if
14:     end while
15:     t ← elapsedTime
16: end while
17: return currentSol
```

specific item). Consequently, different solutions are generated when the procedure is executed repeatedly, and it can be expected that some will be of significantly higher quality than the one obtained employing the deterministic heuristic. In order to efficiently implement these ideas in code, they make use of skewed probability distributions, such as the geometric or the decreasing triangular ones. All in all, the biased randomization approach allows us to obtain multiple solutions based on a deterministic heuristic. As further advancement, our algorithm includes MCS, which is applied during the search to assess the performance of a number of solutions in a stochastic environment by generating scenarios with specific values for each random variable, and computing the expected value of the objective solution (i.e., the expected risk of the portfolio). The open-source quadratic programming solver ojAlgo (http://ojalgo.org), developed in Java, is called to determine the optimal weights allocated to a given set of assets. Additionally, a cache memory (implemented as a hash map data structure) is used in order to avoid calling the solver repeatedly for a specific set of assets.

Details of the proposed methodology

The flowchart diagram of our approach is depicted in Figure B.3 and described next:

1. Consider a SPOP instance defined by n assets. Each asset a_i ($\forall i \in \{1, 2, \ldots, n\}$) has an associated return rate R_i, which is a random variable following a probability distribution, either empirical or theoretical. Each pair of assets a_i, a_j ($\forall i, j \in \{1, 2, \ldots, n\}$) is characterised by a covariance C_{ij}, which is also random and depends on the correlation P_{ij} and the standard deviations S_i and S_j according to the following equation: $P_{ij} = \frac{C_{ij}}{S_i S_j}$

2. Transform the original stochastic problem into a POP instance by means of replacing the random variables by their expected values r_i and $\sigma_i j$.

3. Construct an initial solution ($initSol$) by selecting the k_{min} assets with the highest returns, after including the s assets pre-selected by the investor, and calling the solver. Afterwards, simulation techniques are considered to compute the probability of satisfying the return constraint in the stochastic environment described by the original instance. In particular, a short number of scenarios (sim_{short}) is used to simulate returns. The solution is stored and one moves on to the fourth step, provided the constraint is satisfied. If this is not the case, a feasible solution is searched using a randomised and iterative procedure. First, the pre-selected assets compose a portfolio. In the next step, the non-preselected assets are ordered according to their expected return, and a random number, between $k_{min} - s$ and $k_{max} - s$, are selected using biased randomization, relying on a geometric distribution with a parameter β (Juan et al., 2011a). All weights are set to the minimum value initially, and

then, each weight is set to the maximum value possible (taking into account for an asset a_i the following elements: ε_i, δ_i, and the fraction that remains to be allocated, i.e., $1 - \sum_{i=1}^{n} x_i$) in the order previously established. If an initial solution can be constructed through this, one moves to step 4. It is worthwhile to remark that we avoid using the solver at this step because the focus is on finding an initial feasible solution considering the stochastic environment and not the one with the lowest risk. The time spent searching for a feasible solution is limited by T_{init}, and the algorithm execution stops if no feasible solution is obtained.

4. A list $bestSols$ is created for storing the l best-found solutions in terms of expected risk. Then, $initSol$ is copied into $currentSol$ and k is set to one. Following this, the expected risk of $currentSol$ is computed by using MCS, and the solution is included in the created $bestSols$ list.

5. An iterative procedure is started and steps 6 and 7 are executed during a given amount of time (T_{loop}).

6. A new solution ($newSol$) is created by shaking the current one. This procedure consists of randomly erasing a number of non pre-selected assets in the solution and randomly introducing new assets until reaching k_{max}. The number of assets erased is determined by k. Moreover, a local search is applied to the resulting solution. It aims to improve the solution by replacing the asset with the lowest weight with another one from the list of non-selected assets ordered

7. $newSol$ is compared against $currentSol$. If the former is better in terms of risk associated with the deterministic version of the problem, $newSol$ is considered to be a promising portfolio setting and the return constraint for the stochastic environment is checked for it. In case of being satisfied, the expected risk is computed for the stochastic version of the problem. If the expected risk of $newSol$ is better than that of $currentSol$, then $newSol$ replaces $currentSol$, k is set to one and $bestSols$ is updated. If it is not satisfied, the solution is discarded. If $newSol$ is not better, k is increased in one unit if $k < K$ or set to one otherwise.

8. Once the iterative procedure ends, the algorithm returns $bestSols$. For each solution, a sample of risk measurements is obtained by simulating a large number of scenarios (sim_{large}). We perform a risk analysis where solutions are compared using the distributions of risk. In order to simplify the analysis, it is based on the expected values and the variances of the distributions, and the reliabilities (or probabilities of satisfying the return constraint). Accordingly, the Pareto dominant solutions (i.e., those that are not dominated by another portfolio for one measure while the other measures are at least equally good) are reported to the decision-maker.

Besides being a simple and natural approach, it is also efficient, providing solutions in real time (see Section 5). However, since MCS techniques tend to be time-consuming, their use is minimised by conducting only few runs to assess promising solutions, and a more thorough simulation at the end of the procedure to obtain reliable measures for the decision-makers to base their conclusions on future investments on.

5. Computational experiments

The described algorithm has been implemented as a Java application. A standard personal computer, Intel Core i5 CPU at 3.2 GHz and 4GB RAM with Windows 7 has been used to perform all tests. Our algorithm is executed ten times using different seeds; only the best results are stored. We have experimented with stock market data from the repository ORlib (http://people.brunel.ac.uk/~mastjjb/jeb/orlib/portinfo.html). This set was presented by Chang et al. (2000), and has been largely analyzed (Schaerf, 2002; Armañanzas and Lozano, 2005; Moral-Escudero et al., 2006; Fernández and Gómez, 2007; Di Gaspero et al., 2011). It represents a market index measured at weekly frequency spanning the period from March 1992 to September 1997: Hang Seng (Hong Kong). As suggested by Di Gaspero et al. (2011), the portfolio frontier has been divided into 100 equidistant points on the axis representing the rate of portfolio expected return.

This benchmark instance is deterministic. In order to assess our simheuristic approach, it has been adapted by replacing the deterministic returns and covariances by random variables.

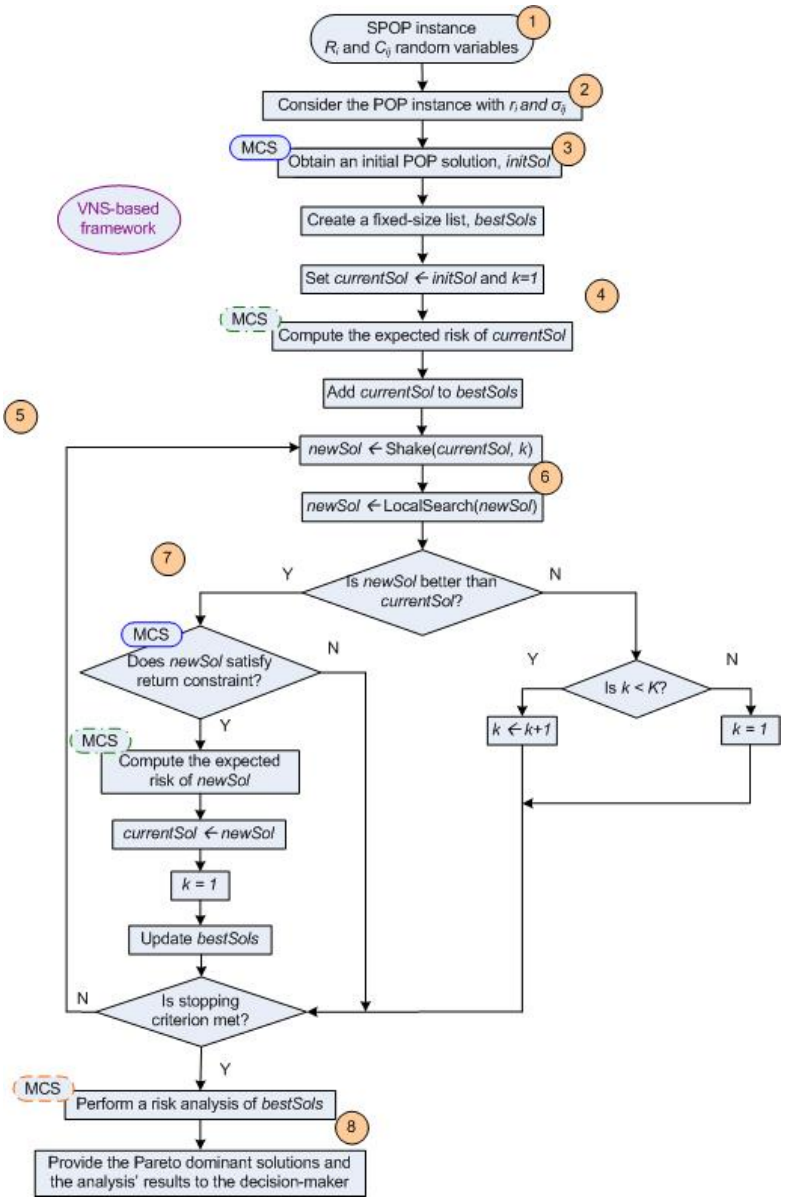


FIGURE B.3: Flowchart of the proposed approach.

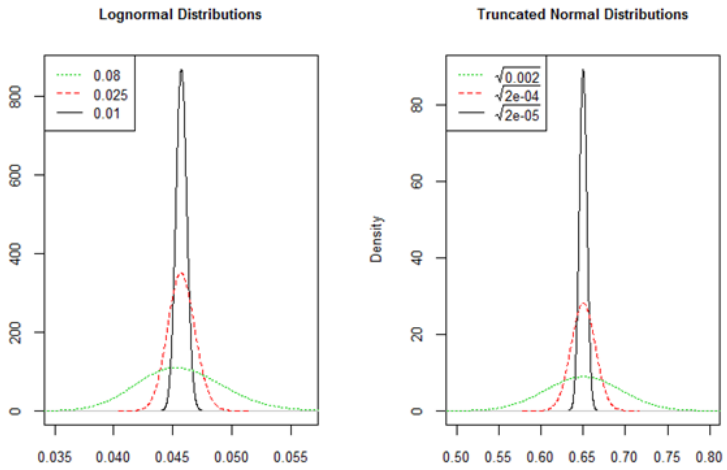


FIGURE B.4: Probability distributions employed.

More specifically, we have considered the following complementary scenarios, which add different degrees of uncertainty on the deterministic POP:

- S_i (Standard deviation) follows a $LN(\mu_S, \sigma_S)$, where LN represents a Log-Normal distribution, and μ_S and σ_S are the mean and the standard deviation of the variable natural logarithm, respectively. They may be determined by the value of the mean and the standard deviation of the variable that are set to σ_i and $c\sigma_i$, being c an input.
- $P_i j$ (Correlation) follows a $TN(\mu_P, \sigma_P, l, u)$, referring TN to truncated Normal distribution, where the parameters are the mean, the standard deviation, and the lower and upper limit, respectively. μ_P is set to the original correlation $\rho_i j$, while σ_P is an input. By the definition of correlation, l and u are set to -1 and 1, respectively. A special case is when $i = j$, then l and u are equal to 1 (i.e., $P_{ij} = 1$).
- R_i follows a $N(\mu_R, \sigma_R)$, where μ_R and σ_R are the mean and the standard deviation of the variable, respectively, which may be determined by the value of the mean and the standard deviation of the variable that are set to r_i and S_i, respectively.

Three values for $c(0.01, 0.025, 0.08)$ and $\sigma_P(\sqrt{0.00002}, \sqrt{0.0002}, \sqrt{0.002})$ have been tested in order to explore three different levels of stochasticity, from lowest to highest. The former values have been selected after performing some quick tests to explore the "reasonable" range for each parameter. Figure B.4 displays the probability distributions of a standard deviation with a mean of 0.0472 (average standard deviation of assets) (left) and those of a correlation with a mean of 0.5562 (average standard correlation among assets) (right). It is worth noting that, being based on simulation, our methodology admits any probability distribution, either theoretical or empirical (in a real-life scenario, empirical data would be employed to find the best fit distribution for each random variable considered). In order to illustrate and analyse our approach, two computational experiments have been carried out. The first experiment considers stochastic covariances (first two scenarios). The second experiment builds on the first one, but additionally introduces stochastic returns (all three scenarios).

The parameter fine-tuning of our algorithm has been performed taking into account suggestions of other authors and results from fast experimental tests. The recommended number of neighbours (K) is 3 (Hansen et al., 2010a). A movement in each neighbour involves changing 25%, 35%, and 45% of the assets, respectively. Regarding the number of solutions stored to analyse at the end (l), a total of 10 are considered. As suggested in Juan et al. (2011a), β is randomly selected from a uniform distribution with parameters 0.05 and 0.25. Finally, sim_{short} and sim_{large} are set to 2500 and 12500, respectively, T_{init} and T_{loop} are set to 5 and 15, respectively.

6. Analysis of results

First experiment: stochastic covariances

Table B.1 summarises the results of the first experiment, where only the covariances are stochastic. The first experiment essentially contrasts and compares two types of solutions: (*i*) the best-found solution to the deterministic version of the problem (BDS); and (*ii*) the best-found solution to the stochastic version of the problem (BSS). In fact, we consider different levels of variability (variance) in the random variables modelling covariances and returns. For each of these variability levels (low, medium, and high) a different stochastic scenario is defined. Notice that portfolio configurations obtained for the deterministic version of the problem can also be used as investment plans for the stochastic version of the problem –even when good solutions for the deterministic version might constitute suboptimal solutions in the stochastic scenario. Put in different words, each portfolio configuration has a different risk measure (cost) for each different environment (deterministic or stochastic). In particular, we are interested in considering different risk measures (costs) associated with the BDS portfolio configuration: the risk measure obtained when employing the BDS in a deterministic scenario, and the expected risk value obtained when using it in each stochastic scenario. To some extend, the former could be considered as a lower bound for the BSS, while the latter could be considered as an upper bound for the BSS. The information gathered in the columns is explained next. The first column reveals the required return, showing only the first and the last 10 values. The next four columns depict the BDS. The following three columns contain the expected risk associated with the BSS for each of the stochastic environments analysed. Also, the average computational time needed for finding the BSS is provided. The last five columns gather some gaps: (*i*) the gap between the risk and the expected risk for a low level of stochasticity of the BDS; (*ii*) the gap between the risk of the BDS and the expected risk of the BSS (also for a low level of stochasticity), being the former a lower-bound of the expected risk of the BDS and the expected risk of the BSS; (*iii*) the gap between the expected risks for the BDS and the BSS considering the environment of a low risk, which quantifies the benefit of using the simheuristic approach instead of assuming constant values; and (*iv*) the previous gap considering the other two environments. Additionally, the average of each ratio has been added at the bottom of the table. Figure B.5 illustrates boxplots of the gaps regarding expected risk between the BDSs and BSSs for the different environments. The expected risk of the BDS is subtracted from that of the BSS so that negative gaps indicate an improvement in the expected risk of the solution. Mean values are represented by diamonds.

Based on these outputs, we may conclude that our algorithm is able to obtain a reasonably good BSS in 0.733 seconds on the average. The gaps between the risk of the BDS and the expected risk of the BDS (when used as a portfolio configuration for the stochastic environment) and the BSS are quite high even for the low-variability scenario (11.82% and 9.92% on the average). As expected, the measure of the BSS is closer to the lower-bound (the risk) than the one of the BDS. Regarding the benefits of using the simheuristic approach in comparison with assuming constant values in terms of expected risk, the mean gaps found for each environment are: -1.68%, -3.09%, and -7.10%. It is important to remark that the gaps are never positive. Thus, the BSS shows a lower expected portfolio variance than the BDS when the latter is used to solve the stochastic version of the problem. Furthermore, the performance of the BDS deteriorates when covariances become more uncertain (i.e., as the variability increases). Intuitively, the BDS can be thought of as a restricted version of the BSS. Such a restriction is costly in terms of the expected portfolio variance, giving rise to a positive variance gap between the BDS and the BSS. Importantly, our research findings are indicative of a potential bias in the solution to a deterministic POP when in fact covariances are stochastic. The size of this bias grows larger when covariances become more uncertain. To correct for this bias, a stochastic POP ought to be solved instead of a deterministic POP.

Second Experiment: Stochastic Covariances and Returns

Results from the second experiment are displayed in Table B.2. As in the previous table, the first column identifies the required return. Columns 2, 3, and 4 detail the expected risk of the BSSs under the lower, medium, and high levels of uncertainty, given a probability of 50% for attaining the required rate of return. Columns 5, 6, and 7 report the gaps between the expected values of risk of the BSSs under the lower, medium, and high levels of uncertainty, when the

TABLE B.1: Hang Seng Stock Market (Hong Kong) with stochastic covariances.

	Deterministic Solution				Stochastic Solutions			Time (s.)	Gaps (%)				
Required Return	Risk (1)	E.R. - Low (2)	E.R. - Med. (3)	E. R. - High (4)	E.R. - Low (5)	E.R. - Med (6)	E.R. - High (7)		(2-1)	(5-1)	(5-2)	(6-3)	(7-4)
0.002861137	0.0006424	0.0007055	0.0007975	0.001114	0.0006948	0.000777	0.0010713	0.211	0.0982	0.0815	-0.0152	-0.0257	-0.0383
0.002941981	0.0006429	0.0007062	0.0008052	0.001195	0.0006952	0.0007772	0.0010701	0.409	0.0985	0.0813	-0.0156	-0.0348	-0.1045
0.003022827	0.0006437	0.0007015	0.0007888	0.0011057	0.0006961	0.0007782	0.0010706	0.627	0.0897	0.0814	-0.0076	-0.0135	-0.0318
0.003103671	0.0006444	0.0007108	0.0008089	0.0011647	0.0006976	0.0007799	0.0010727	0.456	0.1031	0.0826	-0.0186	-0.0358	-0.079
0.003184516	0.0006455	0.0007054	0.0007978	0.00113	0.0006996	0.0007824	0.0010764	0.369	0.0929	0.0838	-0.0083	-0.0192	-0.0474
0.003265361	0.0006468	0.0007057	0.0007951	0.0011208	0.0007011	0.000785	0.0010821	2.03	0.0912	0.084	-0.0065	-0.0127	-0.0346
0.003346206	0.0006483	0.0007113	0.0008062	0.0011714	0.0007021	0.0007857	0.0010806	2.461	0.0972	0.083	-0.0129	-0.0255	-0.0775
0.003427051	0.00065	0.0007143	0.0008095	0.0011649	0.0007035	0.0007869	0.0010809	1.541	0.0989	0.0824	-0.015	-0.0279	-0.0721
0.003507896	0.0006517	0.0007161	0.0008037	0.0010852	0.0007103	0.0007887	0.0010822	5.966	0.0989	0.09	-0.0081	-0.0187	-0.0028
0.00358874	0.0006536	0.0007103	0.000799	0.0011332	0.0007076	0.0007912	0.0010848	0.581	0.0867	0.0826	-0.0038	-0.0098	-0.0427
0.010137479	0.0035774	0.0040554	0.0047712	0.0074653	0.0039666	0.0045855	0.0067181	0.001	0.1336	0.1088	-0.0219	-0.0389	-0.1001
0.010218315	0.0036908	0.004178	0.0049221	0.0076114	0.0040965	0.0047415	0.0069622	0.002	0.132	0.1099	-0.0195	-0.0367	-0.0853
0.010299151	0.0038091	0.0043348	0.0051217	0.0079629	0.0042323	0.0049052	0.0072194	0.001	0.138	0.1111	-0.0236	-0.0423	-0.0934
0.010379986	0.0039324	0.004503	0.0053405	0.0083909	0.0043741	0.0050764	0.0074899	0.001	0.1451	0.1123	-0.0286	-0.0494	-0.1074
0.010460822	0.0040605	0.0046604	0.0055344	0.0087777	0.0045219	0.0052551	0.0077734	0.001	0.1477	0.1136	-0.0297	-0.0505	-0.1144
0.010541657	0.0041937	0.004762	0.0055848	0.0083847	0.0046757	0.0054415	0.0080701	0.001	0.1355	0.1149	-0.0181	-0.0257	-0.0375
0.010622493	0.0043317	0.0048879	0.005757	0.0089222	0.0048354	0.0056353	0.00838	0.001	0.1284	0.1163	-0.0107	-0.0211	-0.0608
0.010703329	0.0044747	0.0051485	0.0061357	0.0096893	0.0050011	0.0058368	0.008703	0.001	0.1506	0.1176	-0.0286	-0.0487	-0.1018
0.010784164	0.0046226	0.0052806	0.006288	0.010186	0.0051728	0.0060458	0.0090391	0.001	0.1423	0.119	-0.0204	-0.0385	-0.1126
0.010865	0.0047755	0.0055134	0.0065833	0.010458	0.005381	0.0063068	0.0096622	0	0.1545	0.1268	-0.024	-0.042	-0.0761
							Average	0.733	0.1182	0.0992	-0.0168	-0.0309	-0.071

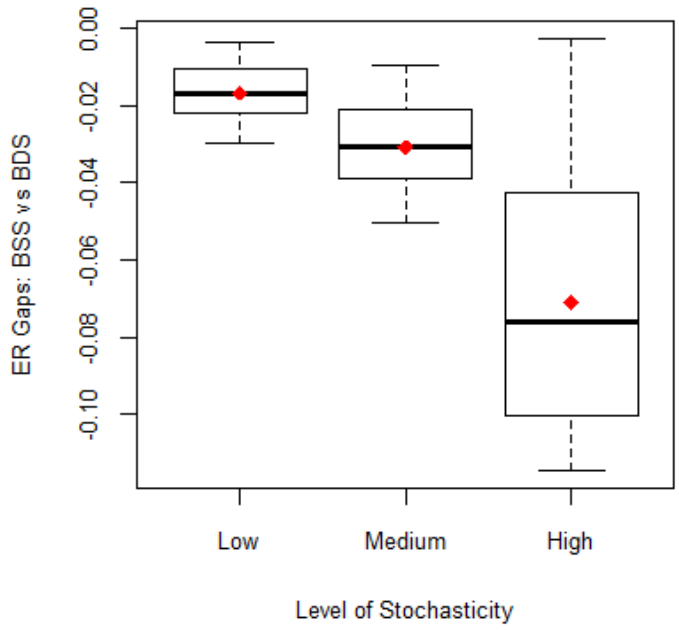


FIGURE B.5: Risk gaps between the best deterministic and stochastic solutions for different levels of stochasticity (environments).

probabilities of attaining the required return are 50% and 47%, respectively (since our benchmarks are extensions of classical ones for the deterministic version of the problem, only some probability values make sense if we wish to keep the feasibility of the optimization problem). Finally, the average computational time is provided. This table also includes the ratio averages (see the last row). Figure B.6 displays the confidence intervals for the mean of expected risk gaps between the solutions for the two probabilities considered. As usual in most studies, a confidence level of 95% is employed to generate the confidence intervals. These confidence intervals are generated from the observations provided by the MCS stage. Also, for illustrative purposes, we have focused on the results associated with a specific required return, 0.0028611366, a probability of 50%, and the environment with a high level of stochasticity. Thus, Figure B.7 shows the confidence intervals for the mean of risks and the reliabilities for the BDS and the three BSSs, i.e, the three best stochastic solutions. Here, the reliability of a given solution refers to the probability that it provides a return higher than the given threshold, and is also obtained during the MCS process.

It can be concluded that the gap between the expected risk of the BSS requiring a probability of 47% and the one with a probability of 50% is relatively small, although it can be relevant and high in some cases. The average values for the different environments are: 0.31%, 0.39%, and 1.84%. Therefore, the gap increases as the level of stochasticity gets higher. From the confidence intervals (Figure B.6), we observe that means are significantly different from zero (i.e., the average gaps are positive), and while they overlap for the first and second environment (although the point estimate is slightly higher for the second), the third is far from that region and does not overlap.

Finally, the last figure displays the confidence intervals for the mean of the risk for the BDS and the three BSSs. The point estimates for the BSSs are clearly lower than that for the BDS, but the difference among the three BSSs are relatively small. The lowest variability (i.e., the narrowest interval) is the associated to the BSS2. Regarding the reliabilities, they are similar too. Overall, the BSS3 could be discarded (is Pareto-dominated), since it has the lowest reliability, a relatively high point estimate, and a wide interval. At the end, the decision-maker should prioritise the different measures to select the best solution according to his/her preferences. Probably, in this case, most decision-makers would choose BSS1, since it offers a low average risk and a high reliability level.

7. Conclusions

This work has addressed a rich (NP-hard) and stochastic variant of the portfolio optimization problem (POP) in which the covariances and the return rates are considered to be random variables instead of constant values as it is usual in the existing literature. A literature review on the

TABLE B.2: Hang Seng Stock Market (Hong Kong) with stochastic covariances and correlations.

Required Return	ER (50%)			ER Gaps [%] (50-47%)			Time (s)
	Low	Medium	High	Low	Medium	High	
0.002861137	0.0007006	0.0007872	0.0011358	0.00%	0.00%	4.38%	1.244
0.002941981	0.0007006	0.0007873	0.001135	0.00%	0.00%	4.44%	2.749
0.003022827	0.0007019	0.0007882	0.0011272	0.00%	0.00%	3.66%	2.647
0.003103671	0.0007027	0.00079	0.0011392	0.00%	0.00%	3.24%	1.7
0.003184516	0.0007068	0.0007925	0.0011075	0.33%	0.00%	1.23%	2.715
0.003265361	0.0007093	0.0007954	0.0011243	0.66%	0.00%	2.15%	1.438
0.003346206	0.0007085	0.000796	0.001137	0.09%	0.19%	2.67%	0.633
0.003427051	0.0007085	0.0007983	0.0011017	-0.11%	0.14%	0.00%	0.764
0.003507896	0.0007138	0.000799	0.0011144	0.54%	0.02%	0.05%	1.365
0.00358874	0.0007161	0.0008014	0.0011068	0.39%	0.00%	0.00%	1.726
0.010137479	0.0040004	0.0046595	0.0071471	0.00%	0.00%	1.05%	1.631
0.010218315	0.0041312	0.0048569	0.0074155	0.00%	0.82%	1.15%	2.8
0.010299151	0.004268	0.0049834	0.0076973	0.00%	0.00%	1.23%	1.421
0.010379986	0.0044359	0.0052031	0.0079926	0.57%	0.90%	1.31%	2.937
0.010460822	0.0045867	0.0053878	0.0083013	0.59%	0.93%	1.37%	4.313
0.010541657	0.0047435	0.0055803	0.0086235	0.61%	0.97%	1.42%	2.31
0.010622493	0.0049063	0.0057805	0.0089592	0.63%	1.00%	1.46%	3.76
0.010703329	0.0050752	0.0059884	0.0093083	0.65%	1.02%	1.49%	1.836
0.010784164	0.0052501	0.006204	0.0096709	0.67%	1.05%	1.52%	0.773
0.010865	0.0054126	0.0063772	0.0098998	0.37%	0.70%	3.06%	1.729
			Average	0.30%	0.39%	1.84%	

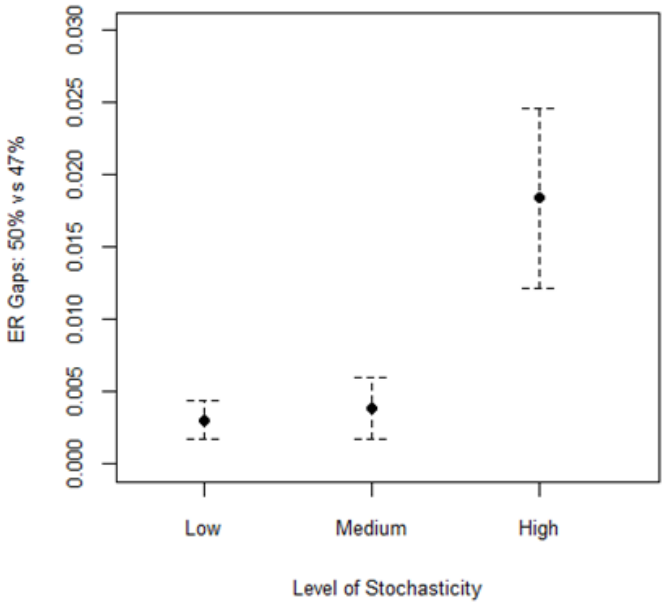


FIGURE B.6: Confidence intervals for the means of expected risk gaps between solutions considering different probabilities.

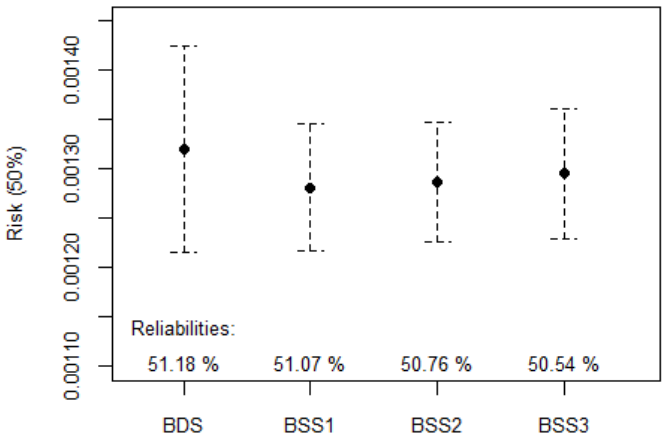


FIGURE B.7: Confidence intervals for the means of risks for the best deterministic and stochastic solutions.

stochastic POP has been provided, which reveals an emerging research topic mainly relying on uncertainty theory where usually only few small illustrative examples are analysed. Then, mathematical formulations for both the rich POP and the rich stochastic POP have been described.

Since real-life financial activities underlie plenty of uncertainty, adding randomness to these elements contributes to diminish the gap between theory and practice. In order to solve the stochastic version of the POP, we have proposed a simheuristic algorithm that combines the VNS metaheuristic (which guides the search of promising solutions) with Monte Carlo simulation techniques. Two computational experiments have been performed to illustrate its use and analyse how the solutions change in terms of expected risk when varying the level of stochasticity, the minimum required return (threshold), and the probability of obtaining a return equal to or greater than that threshold return. Our methodology is able to solve real-sized stochastic instances in small amounts of time. Also, it has been shown that, even in an environment with a relatively low level of variability, a stochasticity-aware (simheuristic) approach may provide much better results than a classical metaheuristic approach generating solutions for the deterministic version of the POP.

Acknowledgements

This work has been partially supported by the Spanish Ministry of Economy and Competitiveness (TRA2013-48180-C3-P, TRA2015-71883-REDT), and FEDER. Likewise, we want to acknowledge the support received by the Department of Universities, Research & Information Society of the Catalan Government (2014-CTP-00001).

References

Adebiyi, A. A. and C. K. Ayo (2015). "Portfolio selection problem using generalized differential evolution 3". In: *Applied Mathematical Sciences* 9.42, pp. 2069–2082.

Armañanzas, R. and J. A. Lozano (2005). "A multiobjective approach to the portfolio optimization problem". In: *IEEE Congress on Evolutionary Computation*. Vol. 2. IEEE Press, pp. 1388–1395.

Babaei, S., M. M. Sepehri, and E. Babaei (2015). "Multi-objective portfolio optimization considering the dependence structure of asset returns". In: *European Journal of Operational Research* 244.2, pp. 525–539.

Behr, P., A. Guettler, and F. Miebs (2013). "On portfolio optimization: Imposing the right constraints". In: *Journal of Banking & Finance* 37.4, pp. 1232–1242.

Bertsimas, D. and A. Thiele (2006). "Robust and data-driven optimization: Modern decision-making under uncertainty". In: *INFORMS tutorials in operations research: models, methods, and applications for innovative decision making* 3, pp. 95–122.

Bianchi, L., D. Marco, L. M. Gambardella, and W. J. Gutjahr (2009). "A survey on metaheuristics for stochastic combinatorial optimization". In: *Natural Computing* 8, pp. 239–287.

Bienstock, D. (1996). "Computational study of a family of mixed-integer quadratic programming problems". In: *Mathematical programming* 74.2, pp. 121–140.

Boussaïd, I., J. Lepagnot, and P. Siarry (2013). "A survey on optimization metaheuristics". In: *Information Sciences* 237, pp. 82–117.

Chang, T.-J., N. Meade, J. E. Beasley, and Y. M. Sharaiha (2000). "Heuristics for cardinality constrained portfolio optimisation". In: *Computers & Operations Research* 27.13, pp. 1271–1302.

Chen, W. (2014b). "An artificial bee colony algorithm for uncertain portfolio selection". In: *The Scientific World Journal* 2014.

Dangi, A. (2013). "Financial Portfolio Optimization: Computationally guided agents to investigate, analyse and invest!?" In: *Master thesis, University of Pune*. Retrieved from http://arxiv.org/abs/1301.4194.

Di Gaspero, L., G. Di Tollo, A. Roli, and A. Schaerf (2011). "Hybrid metaheuristics for constrained portfolio selection problems". In: *Quantitative Finance* 11.10, pp. 1473–1487.

Di Tollo, G. and A. Roli (2008). "Metaheuristics for the portfolio selection problem". In: *International Journal of Operations Research* 5.1, pp. 13–35.

Fernández, A. and S. Gómez (2007). "Portfolio selection using neural networks". In: *Computers and Operations Research* 34.4, pp. 1177–1191.

Fernandez-Viagas, V. and J. M. Framinan (2015c). "NEH-based heuristics for the permutation flowshop scheduling problem to minimise total tardiness". In: *Computers & Operations Research* 60, 27—36.

Hansen, P., N. Mladenovic, and Moreno J. A. (2008a). "Variable neighborhood search". In: *European Journal of Operational Research* 191.3, pp. 593–595.

Hansen, P., N. Mladenovic, and J. A. Moreno Pérez (2008b). "Variable neighborhood search: methods and applications". In: *4OR - A Quarterly Journal of Operations Research* 6, pp. 319–360.

Hansen, P., Mladenović N., Brimberg J., and Moreno J. A. (2010a). "Variable neighborhood search". In: *Handbook of metaheuristics*. Ed. by Gendreau M. and Potvin J.-Y. 2nd. Springer, pp. 61–86.

Huang, X. (2007). "Two new models for portfolio selection with stochastic returns taking fuzzy information". In: *European Journal of Operational Research* 180.1, pp. 396–405.

Juan, A. A., J. Faulin, J. Jorba, D. Riera, D. Masip, and B. Barrios (2011a). "On the use of Monte Carlo simulation, cache and splitting techniques to improve the Clarke and Wright savings heuristics". In: *Journal of the Operational Research Society* 62, pp. 1085–1097.

Juan, A. A., H. R. Lourenço, M. Mateo, R. Luo, and Q. Castella (2014f). "Using iterated local search for solving the flow-shop problem: parallelization, parametrization, and randomization issues". In: *International Transactions in Operational Research* 21.1, pp. 103–126.

Juan, A. A., J. Faulin, S. Grasman, M. Rabe, and G. Figueira (2015a). "A review of simheuristics: extending metaheuristics to deal with stochastic optimization problems". In: *Operations Research Perspectives* 2, pp. 62–72.

Kolm, P. N., R. Tütüncü, and F. J. Fabozzi (2014). "60 years of portfolio optimization: Practical challenges and current trends". In: *European Journal of Operational Research* 234.2, pp. 356–371.

Liu, Y., X. Wu, and F. Hao (2012a). "A new chance–variance optimization criterion for portfolio selection in uncertain decision systems". In: *Expert Systems with Applications* 39.7, pp. 6514–6526.

Mansini, R., W. Ogryczak, and M. G. Speranza (2014). "Twenty years of linear programming based portfolio optimization". In: *European Journal of Operational Research* 234.2, pp. 518–535.

Maringer, D. (2005). *Portfolio management with heuristic optimization*. Berlin: Springer.

Markowitz, H. (1952). "Portfolio selection". In: *The journal of finance* 7.1, pp. 77–91.

Mladenović, N. and P. Hansen (1997). "Variable neighborhood search". In: *Computers & Operations Research* 24.11, pp. 1097–1100.

Moral-Escudero, R., R. Ruiz-Torrubiano, and A. Suárez (2006). "Selection of optimal investment portfolios with cardinality constraints". In: *2006 IEEE International Conference on Evolutionary Computation*. IEEE, pp. 2382–2388.

Moreno-Vega, J. M. and B. Melián (2008). "Introduction to the special issue on variable neighborhood search". In: *Journal of Heuristics* 14.5, p. 403.

Nazemi, A., B. Abbasi, and F. Omidi (2015). "Solving portfolio selection models with uncertain returns using an artificial neural network scheme". In: *Applied Intelligence* 42.4, pp. 609–621.

Nguyen, T. T., L. Gordon-Brown, A. Khosravi, D. Creighton, and S. Nahavandi (2015). "Fuzzy portfolio allocation models through a new risk measure and fuzzy Sharpe ratio". In: *IEEE Transactions on Fuzzy Systems* 23.3, pp. 656–676.

Ning, Y., H. Ke, and Z. Fu (2015). "Triangular entropy of uncertain variables with application to portfolio selection". In: *Soft Computing* 19.8, pp. 2203–2209.

Qin, Z. (2015). "Mean-variance model for portfolio optimization problem in the simultaneous presence of random and uncertain returns". In: *European Journal of Operational Research* 245.2, pp. 480–488.

Sawik, B. (2012a). "Bi-criteria portfolio optimization models with percentile and symmetric risk measures by mathematical programming". In: *Przeglad Elektrotechniczny* 88.10B, pp. 176–180.

Schaerf, A. (2002). "Local search techniques for constrained portfolio selection problems". In: *Computational Economics* 20.3, pp. 177–190.

Soleimani, H., H. R. Golmakani, and M. H. Salimi (2009). "Markowitz-based portfolio selection with minimum transaction lots, cardinality constraints and regarding sector capitalization using genetic algorithm". In: *Expert Systems with Applications* 36.3, pp. 5058–5063.

Talbi, E.-G. (2009). *Metaheuristics: From design to implementation*. New Jersey: John Wiley & Sons.

Zhang, W.-G., Y.-J. Liu, and W.-J. Xu (2012). "A possibilistic mean-semivariance-entropy model for multi-period portfolio selection with transaction costs". In: *European Journal of Operational Research* 222.2, pp. 341–349.

B.2 Statistical and machine learning approaches for the minimization of trigger errors in parametric earthquake catastrophe bonds

Laura Calvet[1], Madeleine Lopeman[2], Jésica de Armas[1], Guillermo Franco[2], Angel A. Juan[1]

1. Computer Science Department, IN3, Open University of Catalonia, Castelldefels, Spain.
e-mail: {lcalvetl, jde_armasa, ajuanp}@uoc.edu
2. Guy Carpenter & Company, LLC, Dublin, Ireland.
e-mail: {Madeleine.Lopeman, Guillermo.E.Franco}@guycarp.com

Abstract

Catastrophe (CAT) bonds are financial instruments designed to transfer risk of monetary losses arising from earthquakes, hurricanes, or floods to the capital markets. The insurance and reinsurance industry, governments, and private entities employ them frequently to obtain coverage for large losses. Parametric CAT bonds constitute a special sub-class that bases its payments on certain physical features of events. For instance, given certain event parameters such as magnitude of the earthquake and the location of its epicenter, the bond may pay a fixed amount or not pay at all, according to a binary outcome. This paper reviews eight statistical and machine learning techniques for classification of events in order to design a trigger mechanism that identifies which events should produce bond payments while minimizing basis risk (trigger error). Use of these techniques relies on training data from CAT models containing large samples of simulated earthquakes that characterize the seismic risk of a given region. Although the numerical accuracy and the computational performance of all methods is very high, non-linear techniques provide better results. Several lines of future research based on parametric trigger mechanisms are discussed.

Keywords: catastrophe bonds, risk of natural hazards, classification techniques, earthquakes, insurance.

1. Introduction

Catastrophe (CAT) bonds are financial instruments that package catastrophe risk in a tradeable security. These tools are in effect responsible for the existence of a new market for trading risk at the frontier between finance and insurance, the so-called convergence market (Cummins and Weiss, 2009), which promises an enormous supply of capital for CAT risk transfer as long as pricing remains attractive for all parties involved. By purchasing a CAT bond, investors take the risk from a sponsor (risk ceding party) in exchange for some interest or spread. This spread constitutes the "premium" that compensates the risk-taking party.

CAT bonds can be of different types depending on how their payment behavior is structured. Earthquake CAT bonds in particular can base their payments on a variety of proxies (Wald and Franco, 2017). While some base payments on actual, experienced losses (indemnity), others (parametric) base them on the observable and measurable parameters that describe an event. Strategies to provide coverage for large losses ensuing after earthquakes through these parametric tools have been in use since the 1990s (Franco, 2014). These instruments have allowed insurers, reinsurers, governments, private entities and catastrophe pools to cede risks of earthquake losses to the capital markets via transparent mechanisms associated with physical event features. Since they bypass the claims adjusting process, these tools provide a very fast recovery of funds to their sponsor after an event. Their popularity in the market is due to historically lower prices relative to traditional (re)insurance and their appeal among investment and hedge funds is due to their transparency. Lately, as traditional reinsurance pricing has decreased significantly, the price differential between traditional and alternative risk transfer (sometimes referred to as ART) is very small and is no longer the driving rationale for seeking parametric coverage. Rather, sponsors now look to parametric risk transfer for the flexibility and the ease of payment it provides.

Parametric earthquake CAT bonds employ a kind of trigger mechanism, typically a numerical check of some sort, to determine the payment that should take place when an earthquake occurs. These trigger mechanisms rely on obtainable physical characteristics of the event via respected

third parties, often public agencies (Cummins, 2007; Croson and Kunreuther, 1999). Since neither the investor nor the sponsor has the ability to manipulate this information, the risk transfer process is without moral hazard (the risk that the parties involved influence the payment outcome). Within the realm of parametric earthquake CAT bonds there are also several classes of tools. Some, first-generation parametric CAT bonds or so-called "CAT-in-a-box" triggers, rely on the main physical descriptors of an earthquake event (see for instance Cardenas et al., 2007; Franco, 2010; Franco, 2013). Others, second-generation indexes, rely on spatially-distributed features such as ground motions recorded at sensors located throughout a region (see for instance Goda, 2013; Goda, 2014; Pucciano et al., under review).

Earthquakes around the world cause enormous losses, of which only about 30% have insurance coverage (Guy Carpenter, 2014). These financial impacts often disrupt individual livelihoods and national economies. Therefore, the possibility of expanding the coverage of insurance to minimize these impacts is very appealing. Making earthquake insurance more accessible, however, is difficult for traditional providers since their operations are typically resource- and time-consuming. Parametric risk transfer, in contrast, can be seamless, fast and cheap but in order to be viable, parametric solutions need to be accurate. They also need to be designed and customized without much effort so they can be easily industrialized.

This work focuses on exploring strategies from statistical and machine learning approaches to design trigger mechanisms accurately and quickly. We review eight techniques to classify events as to whether they should trigger a payment or not, following a binary payment scheme often used in the industry. Events are classified using the four fundamental parameters of focal location (longitude, latitude and depth) and moment magnitude. The characteristics of the trigger mechanism are introduced in detail in Section 2.

The eight techniques used for classification are the Nearest Neighbors Classifier, the Naïve Bayes Classifier, Linear/Quadratic Discriminant Analysis, Classification Trees, Logistic Regression, Clusterwise Logistic Regression, Neural Networks, and Support Vector Machines. These techniques are all introduced in Section 3, where we present a brief description of each of them. Note that all the approaches need to be trained with a given dataset. These data need to constitute a large sample of events and need to include a monetary loss for each earthquake. Therefore, we turn to an earthquake CAT model to obtain a viable training dataset since historical catalogs usually do not contain a large enough sample of this type of information.

CAT Models have been discussed in previous studies (e.g. Grossi and Kunreuther, 2005) and we will not discuss the CAT modeling process in detail in this paper. It is sufficient to realize that an earthquake loss model (such as GEM's OpenQuake for example) can be used to produce a dataset containing a large amount of simulated earthquakes in harmony with the local seismic setting. These records should include the four main physical parameters enumerated before and the corresponding simulated loss. For each synthetic earthquake event in the catalog, the model computes a ground motion footprint, which is in turn translated into estimated levels of damage to a user-defined portfolio of properties distributed in space. The target of the classifier algorithms, in short, is to discriminate events based on their physical parameters to identify large loss-producing events. This will be discussed in detail in the following sections.

In order to test the performance of these techniques against a known benchmark, we recapture the analysis presented in Franco (2010) and solve the same problem again in Section 4, using the same dataset, with each of the new approaches. We then compare across methods on such issues as accuracy, computational effort, and spatial correlation of the classifier results. Our conclusions and suggestions for future research are collected in Section 5.

2. The Trigger Mechanism

Consider a set of l earthquake events in a geographic region of interest A. An earthquake event i is characterized by a magnitude m_i, a hypocenter depth d_i, and epicenter coordinates (x_i, y_i) within A. A binary trigger will determine whether a payment should be disbursed due to event i. This response is represented by the variable B', whose values 1/0 indicate trigger/no-trigger (payment/no-payment). Two situations may arise: (1) at least one earthquake i triggers the bond ($B'_i = 1$) during its contract life, which means that the entire bond principal has to be disbursed and, as a consequence, the buyers of the bond lose their investment (and the bond sponsors receive compensation), or (2) no earthquake triggers the bond during its life, in which case the principal is returned to the investors with interest.

Since the payment of a large sum of money is at stake, it is important that the trigger performs as desired, i.e. that the trigger responds positively to events that cause a large loss beyond a design threshold and that it does not respond for events that cause a loss below this threshold. The accuracy of the trigger determines its success in the market. Triggers that behave erratically erode the confidence of the markets in these tools and therefore jeopardize the risk transfer process. It is crucial to design triggers that behave as they should.

To describe the accuracy of the trigger, first consider a reference variable B that represents its idealized behavior and that depends on a measure based on the losses (typically monetary). For an earthquake event i, this variable can be described as follows:

$$B_i = \begin{cases} 0 & \text{if } L_i \leq L \\ 1 & \text{otherwise} \end{cases} \tag{B.1}$$

where L_i is the actual loss caused and L is a loss threshold specified by the sponsor, usually expressed in terms of a specific return period. In this idealized scenario, events trigger this CAT bond only if the corresponding loss is above a given pre-specified threshold L.

The objective of parametric trigger mechanism design is to develop a classification mechanism that uses physical parameters of events to determine the trigger behavior B'. Discrepancies between variables B and B' or the sum of errors ($E = \sum_{i=1}^{l} I(B_i = B'_i)$), represent lack of correlation between the output of the trigger and the ideal trigger. Effective parametric trigger mechanism design aims to minimize these discrepancies.

A database including a set of events, their characteristics and the variable B can be used to calculate trigger errors for this specific set of events. A measure of the loss has to be obtained or estimated to compute B. It is preferable to have a reliable historical dataset including a high number of events but in earthquake research, this is not possible due to the low frequency of earthquakes and the great uncertainty surrounding their associated losses. For this reason, the design of triggers for seismic risk relies on simulated CAT model output.

According to the description offered in this section, the development of a trigger mechanism can be labelled as a binary classification problem, allowing us to employ a wide range of techniques to address it. In the following sections, some of them are introduced and tested, and their use is illustrated.

3. Statistical and Machine Learning Approaches

Classification techniques (Kotsiantis, 2007) constitute a set of procedures from statistics and machine learning (more specifically, supervised learning) to determine a category or class for a given observation. Having a dataset of l observations composed of explanatory or independent variables $(X_1, X_2, ..., X_n)$, and a response or dependent variable Y, these techniques attempt to explain the relationships between variables and/or classify new observations based on the explanatory variables.

Nowadays, there are plenty of classification techniques. Some of the most employed, e.g., Linear Discriminant Analysis or Logistic Regression, have been applied for more than five decades. These are mainly linear methods. Boosted by the computing advances in the 1980s and 1990s, non-linear methods such as Classification Trees, Neural Networks and Support Vector Machines emerged and/or started to attract attention.

This section introduces some well-known and powerful techniques that we propose to automatically design a trigger. The reader interested in comprehensive and practical descriptions is referred to the books written by Hastie et al. (2009) and Lantz (2013).

The Nearest Neighbors Classifier

The Nearest Neighbors classifier is a simple technique that assigns a new observation to the class of the most similar observations, so-called neighbors. Therefore, it is suitable when observations of the same class tend to be homogeneous. Its main weaknesses are: not producing a model (which hinders the exploration of relationships among variables), taking a relatively high amount of time, and consuming a large amount of memory. This classifier depends on a parameter k representing the number of neighbors. The neighbors are selected according to a distance function, usually Euclidean. This parameter allows the balance between overfitting and underfitting (also known as bias-variance trade-off): a large k reduces the variance caused by noisy data or outliers but may ignore small/local patterns; conversely, a small value may introduce too bias.

The Naïve Bayes Classifier

The naïve Bayes classifier is based on Bayes' theorem. "Naïve" refers to the assumption that all variables are independent and equally important. Even if this condition is not usually met in real-life applications, this classifier frequently provides competitive results. The posterior probability for a given class y is computed as:

$$P(Y = y \mid X_1 = x_1 \cap X_2 = x_2 \cap ... \cap X_n = x_n) = \frac{P(X_1 = x_1 \mid Y = y)P(X_2 = x_2 \mid Y = y)...P(X_n = x_n \mid Y = y)P(Y = y)}{P(X_1 = x_1)P(X_2 = x_2)...P(X_n = x_n)} \quad \text{(B.2)}$$

The classification for a given observation is obtained by comparing the probabilities of each class given the values of the explanatory variables, and selecting the class associated to the highest probability. There are many classifiers differing in the assumption made regarding the distribution of $P(X_j = x_j \mid Y = y)$. Gaussian distributions constitute a typical choice. This technique employs frequency tables and, consequently, each variable must be categorical. Numeric variables are usually discretized.

Linear and Quadratic Discriminant Analyses

In Linear Discriminant Analysis, the distribution of the explanatory variables is separately modeled in each of the classes, and then the Bayes' theorem is used to flip these around into estimates for the probability of the response variable taking a specific value given the explanatory variables. Commonly, these distributions are assumed to be Gaussian. In this case, the resulting models are similar to those provided by Logistic Regression. Linear Discriminant Analysis is more commonly employed when there are more than two classes. While this technique assumes that observations are drawn from a distribution with a common covariance matrix in each class (which leads to linear decision boundaries), Quadratic Discriminant Analysis does not make assumptions on the covariance matrix (producing quadratic decision boundaries).

Classification Trees

Contrary to global models (where a predictive formula is supposed to hold in the entire data space) such as those of Logistic Regression, Classification Trees try to partition the data space into small enough parts where a simple model can be applied. The results can be represented as a tree composed of internal and terminal (or leaf) nodes, and branches. Its non-leaf part is a procedure to determine for each observation which model (i.e., terminal node) will be used to classify it. At each internal node of the tree, the value of one explanatory variable is checked and, depending on the binary answer, the procedure continues to the left or to the right sub-branch. A classification is made when a leaf is reached.

The most relevant advantage of this classifier is the easiness to understand what trees represent. They mirror human decision-making more closely than other techniques. Furthermore, trees require little data preparation, are able to handle both numerical and categorical data, and perform well (i.e., use standard computing resources in reasonable time) with large datasets.

Logistic Regression

Logistic Regression techniques are designed to model the posterior probabilities of each class by means of linear functions. These probabilities, such as the one shown below, must be non-negative and sum to one.

$$P(Y = y \mid X_1 = x_1 \cap X_2 = x_2 \cap ... \cap X_n = x_n) = \frac{exp(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_n x_n)}{1 + exp(\beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_n x_n)} \quad \text{(B.3)}$$

These models are usually fit by maximum likelihood employing Newton's method. The previous expression can be rewritten in terms of log-odds as follows:

$$log\Big(\frac{P(Y = y \mid X_1 = x_1 \cap X_2 = x_2 \cap ... \cap X_n = x_n)}{1 + P(Y = y \mid X_1 = x_1 \cap X_2 = x_2 \cap ... \cap X_n = x_n)}\Big) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_n x_n \quad \text{(B.4)}$$

This technique is especially useful when the aim is to explain (i.e., not only classify) the outcome based on the explanatory variables. Non-linear functions can be considered including interactions and transformations of the original variables.

Clusterwise Logistic Regression

While Regression Analysis consists of fitting functions to analyze the relationship between variables, Clustering seeks subsets of similar observations (or variables) in a dataset. Thus, the aim of Clusterwise Regression is to combine both techniques in order to discover trends within data when more than one trend is likely to exist (DeSarbo and Cron, 1988). This technique is highly flexible because different functions can be estimated. It is considered a "white-box technique" in that its mathematical systems are not complex and its results are relatively easy-to-interpret.

Neural Networks

Neural Networks model the relationship between the explanatory variables and the response variable using a model inspired by how a biological brain responds to stimuli from sensory inputs. They extract linear combinations of the explanatory variables as derived variables and model the response variable as a non-linear function of these transformed variables. These models have several kinds of layers: the input layer, the output layer, and one or more hidden layers between them. Each layer contains neurons representing the variables. Increasing the number of hidden layers and/or neurons adds complexity and may improve computational capacity. With too few layers, the model may lack the flexibility to capture non-linearities in data. Neural Networks tend to have many weights, which can cause problems of overfitting. Weight decay is a method of regularization to prevent it. The "backpropagation" algorithm is a commonly-employed technique for parameter estimation or training a Neural Network.

Support Vector Machines

A Support Vector Machine can be imagined as a surface that defines a boundary between various points of data that represent observations plotted in a multidimensional space. The goal is to create a flat boundary, called a hyperplane, which leads to fairly homogeneous partitions of data on either side. Among all potential hyperplanes, the one that creates the greatest separation between classes (a soft margin may be considered for the case on non-linearly separable data) is selected. The support vectors are the points from each class that are the closest to the hyperplane; each class must have at least one. In many real-life applications, the relationships between variables are non-linear. A key feature of this technique is its ability to efficiently map the observations into a higher dimension space by using the kernel trick. As a result, a non-linear relationship may be transformed into a linear one.

Discussion of Classification Techniques

Several techniques have been presented in the literature to design trigger mechanisms that determine –from an earthquake's physical characteristics– whether a principal bond should be paid (Franco, 2010; Franco, 2013). As mentioned, the aim of this work is to introduce and illustrate the application of simple, well-known, and efficient techniques that have heretofore not been explored in this context.

Neural Networks and Support Vector Machines constitute two relatively modern and powerful techniques. Typically, they are able to reach high levels of accuracy by capturing non-linear relationships between variables. However, this same characteristic makes them prone to overfitting. There are many procedures to avoid this problem such as the addition of a parameter to limit the growth of the weights or the introduction of randomness into the training data or the training algorithm. Sometimes it may be difficult to avoid overfitting and underfitting. Training Neural

Networks often takes a long time, and both techniques require a non-trivial process of fine-tuning parameters. Furthermore, the resulting models are difficult if not impossible to interpret. For this reason, application of these techniques is almost always limited to classification/prediction purposes.

Techniques such as Nearest Neighbors and Naïve Bayes Classifiers are easier to understand and implement and may provide relatively high accuracy. While the first is non-parametric and, consequently, flexible or unstable, the second relies on some assumptions that may be quite unrealistic in most cases.

Logistic Regression is a well-established technique, which enables the understanding of the effects of the explanatory variables on the response. Clusterwise Logistic Regression aims to incorporate the strengths of Logistic Regression while offering more flexibility, which should lead to a better understanding of the relationships among variables and higher accuracy. Classification Trees constitute an efficient technique that only uses the most important variables and results in a logic model. As other techniques studying non-linear relationships, these three techniques are particularly susceptible to overfitting or underfitting the model. Typically, small changes in training data may lead to significant modifications. In addition, Classification Trees may derive decisions that seem counterintuitive or are unexpected.

Closely related to Logistic Regression, the classic Linear/Quadratic Discriminant Analysis techniques search for the linear/quadratic combination of variables that explains the data the best. Logistic Regression is preferred if the assumption of normally-distributed explanatory variables does not hold. Otherwise, Discriminant Analysis can provide better results.

All these techniques have different features worthy of consideration when addressing a classification problem. Consequently, all are included in the following computational experiments.

4. Computational Experiments

This section illustrates the application of the techniques introduced in Section 3 and compares the results with those obtained in Franco (2010). A framework for evaluation is presented such that the techniques can be compared to one another and to the reference methodology along the dimensions of accuracy, efficiency, and spatial correlation.

The dataset analyzed is an earthquake catalog representing 10,000 years of simulated seismic activity in and around Costa Rica. The catalog contains a total of 24,957 earthquakes, and a more detailed description can be found in the aforementioned work. The threshold for identifying triggering events corresponds to the 100-year return period loss.

4.1 Evaluation Framework

In the case of parametric trigger design, it is difficult a priori to select the “best" classification technique for two main reasons. First, it is a multi-objective problem. Although from a statistical perspective, the sole objective may be to maximize accuracy, in real-life applications many other characteristics will likely play an important role. These may include ease of implementation, ease of explanation to non-experts, popularity, and existence of graphical representations or summaries of the outputs, among many others. The second reason is that, assuming we are only interested in the accuracy, the best technique will depend on the data at hand. Consequently, we present a general discussion of all techniques, and evaluate the trigger mechanisms they produce in three ways.

First, the confusion matrix (Table B.1) is obtained for each trigger mechanism. This table summarizes the alignments and discrepancies between the behavior of the designed trigger mechanism and the idealized trigger behavior (described in Section 2). In the context of parametric triggers, B' is a function representing the predicted trigger behavior and B is a function representing the idealized trigger behavior. In both cases, the function is equal to 1 if the bond triggers and is equal to 0 otherwise.

Table B.1: Structure of a confusion matrix

		Predicted Class	
		$B' = 0$	$B' = 1$
Idealized Class	$B = 0$	True Positive (TP)	False Positive (FP)
	$B = 1$	False Negative (FN)	True Negative (TN)

Next, several metrics are computed from the confusion matrix to quantify performance of each technique's trigger mechanism: error, sensitivity, and specificity. The formula for computation thereof are shown below.

$$Error = \frac{FP + FN}{TP + FP + FN + TN} \tag{B.5}$$

$$Sensitivity = \frac{TP}{TP + FN} \tag{B.6}$$

$$Specificity = \frac{TN}{TN + FP} \tag{B.7}$$

Both false positive and false negative are equally penalized in this framework. In other words, we simply focus on minimizing the total number of errors. The Error metric above quantifies the rate at which the trigger mechanism misclassifies events. [1] Sensitivity characterizes how often the mechanism triggers when it should trigger, and specificity characterizes how often the mechanism does not trigger when it should not. The time required to design the trigger mechanism is also reported for each technique. The metrics described above constitute the numerical evaluation of the trigger mechanisms, and are presented in Section 4.4. Moreover, maps of the resulting trigger patterns are produced for a subset of techniques. This exercise is intended to assess whether classification techniques produce trigger mechanisms with realistic geospatial trigger patterns.

4.2 Application of Classification Techniques

As mentioned in Section 2, the design of a parametric trigger mechanism is driven by the minimization of discrepancies between its outputs and those from a trigger with an idealized behavior (one based directly on the losses). If the resulting trigger mechanism is expected to be useful for new or unseen observations, one should avoid employing the same observations for developing the mechanism and assessing its performance. This could lead to a problem of overfitting (i.e., obtaining complex models that capture specificities of the data but do not generalize well for other observations). An effective technique to avoid this problem is to split the dataset into three subsets: a training set used for constructing the triggers, a validation set employed to tune the parameters, and a test set required to assess their performance. We apply this approach using 50% of the observations for training, 25% for validation, and the remaining 25% for testing. z-score standardization has been applied for all techniques except Classification Trees, Logistic Regression and Clusterwise Logistic Regression. A confidence level of 95% has been considered for the statistical tests. Details of the application of each of the classification techniques are provided in the following paragraphs. The R program (R Core Team, 2012) has been used.

The Nearest Neighbors Classifier. This technique requires a choice of the number of nearest neighbors to consider. Values ranging from 3 to 10 have been tested, and the corresponding accuracies associated to each value have been assessed using the validation set. Ultimately, 5 nearest neighbors are considered for construction of the trigger mechanism, since this provides the highest accuracy but is still small enough to reduce both the variance and the computational time required to make predictions.

Classification Trees. Construction of a Classification Tree relies on selection of the complexity parameter (a parameter to measure the tree cost-complexity). Values from 0.01 up to 0.20 have been tested, and the most accurate results correspond to the value 0.05. Figure B.1 shows the final tree representation. Observations which satisfy the condition shown for each internal node terminate to the left, otherwise, they proceed to the right. The percentage shown at the bottom of each node indicates the proportion of observations that reach that node. The value above that percentage refers to the binary classification.

[1] Note that error is equal to one minus accuracy.

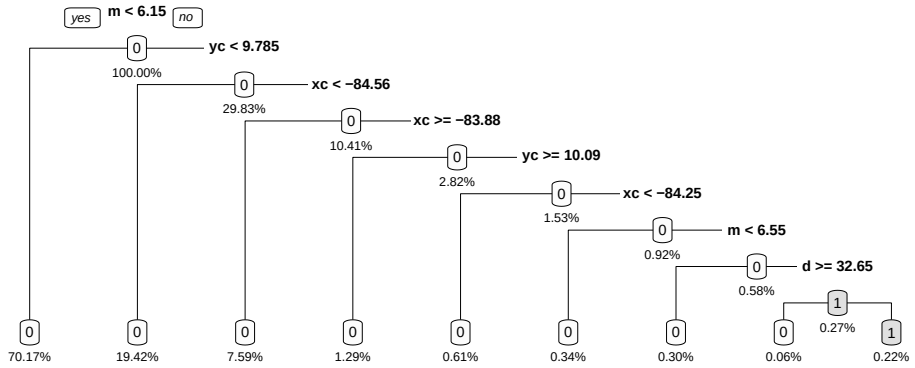


FIGURE B.1: Classification Tree

Neural Networks. Even if complex and powerful Neural Networks exist, we focus on a topology characterized by only one hidden layer. Despite its minimalism, this approach is commonly used, tends to provide good results and is conceptually simple. The number of units in the hidden layer (26) has been tuned by testing the set of values ranging from 10 to 40.

Support Vector Machines. In order to efficiently employ this technique, it is required to select a kernel and tune the corresponding parameters. The most popular kernels have been considered and are shown in Table B.2. There is also a parameter related to the cost of a misclassification for which the following values have been considered: $0.01, 0.1, 1, 5$, and 10. The results reveal that the best option is a polynomial kernel with the following parameters: $cost = 10$, $\alpha = 0.4$, $c = 0.4$, $d = 4$.

TABLE B.2: Kernels considered for Support Vector Machines

Linear	$k(a,b) = a^T b$
Polynomial [a]	$k(a,b) = (\alpha a^T b + c)^d$
Radial Basis [b]	$k(a,b) = exp(-\gamma \lvert a-b \rvert^2)$
Sigmoid [c]	$k(a,b) = tanh(\sigma a^T b + e)$

[a] Values tested for α, c, and d, respectively: $\{0.1, 0.2, 0.3, 0.4\}$, $\{0, 0.2, 0.4, 0.6\}$, and $\{2, 3, 4, 5\}$.
[b] Values tested for γ : $\{0.1, 0.2, 0.3, 0.4\}$.
[c] Values tested for σ and e, respectively: $\{0.1, 0.2, 0.3, 0.4\}$ and $\{0, 0.2, 0.4, 0.6\}$.

4.3 External Validation

In order to validate the application of these techniques to the development of parametric triggers for earthquake catastrophe bonds, we compare our results with those provided by the methodology in Franco (2010). In this paper, the author proposes the construction of binary "cat-in-a-box" trigger mechanisms, where the geographical space is discretized in square boxes or subregions of the same size. Each sub-region belongs to a specific zone denoted as k. This approach relies on the concept of optimization and its aim is to determine the parameters of a trigger mechanism for each zone as well as the zone assignment of each sub-region such that the total trigger error is minimized. Concretely, the trigger mechanism for zone k has the following structure:

$$\forall (x_i, y_i) \in A_k, \qquad B'_i = \begin{cases} 0 & \text{if } m_i \leq M_k \quad \text{or } d_i \geq D_k \\ 1 & \text{if } m_i \geq M_k \quad \text{or } d_i \leq D_k \end{cases} \tag{B.8}$$

where M_k and D_k represent the parametric triggers for the zone, namely the magnitude and depth thresholds, respectively. All sub-regions belonging to zone k have the same trigger structure. An Evolutionary Algorithm (EA) is implemented to address this optimization problem and is executed for different combinations of geographic resolution and number of zones. Although the paper does not report computational times, these methods may consume several hours to perform the parameter optimization.

4.4 Performance

The performance of the trigger mechanisms designed using all nine statistical and machine learning techniques and using the EA employed in Franco (2010) is reported and discussed here. Performance measures are shown in Table B.3. Total time takes into account the time to construct the trigger, fine-tune its parameters and test its performance.

TABLE B.3: Parametric trigger mechanism performance for ten techniques.

Technique	Error	Sensitivity	Specificity	Time (sec.)
Nearest Neighbors classifier	0.18%	99.84%	**94.44%**	7.22
Naïve Bayes classifier	0.77%	99.58%	4.35%	1.62
Linear Discriminant Analysis	0.64%	99.57%	0.00%	0.28
Quadratic Discriminant Analysis	0.42%	99.63%	57.14%	**0.12**
Classification Trees	0.24%	99.79%	87.50%	2.62
Logistic Regression	0.45%	99.58%	33.33%	0.87
Cluster-wise Logistic Regression	0.43%	99.57%	undefined	5.7
Neural Networks	**0.14%**	**99.94%**	82.14%	190.86
Support Vector Machines	0.27%	99.78%	81.25%	161.25
Evolutionary Algortihm (Franco, 2010)	0.34%	99.86%	55.56%	hours

A suitable trigger mechanism design should exhibit low error and high specificity and sensitivity and should require minimal computational effort. It can be concluded from the table that the non-linear and non-parametric techniques obtain the best performances of the statistical techniques in terms of accuracy, sensitivity and specificity. In particular, Nearest Neighbors classifier, Classification Trees, Neural Networks and Support Vector Machines are all consistently superior across the three metrics. The results reveal a high variability with respect to computational time, ranging from a few seconds to several minutes. There tends to be trade-off between accuracy and time-required, particularly in the cases of Neural Networks and Support Vector Machines, both of which require significantly more time than the other techniques.

Several techniques exhibit superior performances to the EA in terms of accuracy, sensitivity and specificity. While EA produces relatively low error rates, the time required is significantly longer than all of the statistical and machine learning techniques.

The triggering events in the idealized trigger mechanism (those for which $B = 1$) comprise less than 0.5% of the total test catalog, while the other 99.5% of catalog events do not trigger the idealized bond. Hence, a supposed "null" trigger mechanism in which no events ever trigger the bond would exhibit 99.5% accuracy (0.5% error), 100% sensitivity and 100% specificity. The burden in this case is therefore on any designed trigger mechanisms to outperform this null trigger mechanism benchmark. Eight out of the ten techniques produce trigger mechanisms superior to the null trigger mechanism in terms of accuracy, while the Naïve Bayes Classifier and Linear Discriminant Analysis perform worse by a small margin.

That so few events trigger the bond in the idealized scenario suggests that a larger catalog might produce more informative and nuanced results using the statistical and machine learning techniques for parametric trigger mechanism design. With a larger catalog to "learn" from, the techniques would have more triggering events from which to decipher patterns and connections. Reduction of the loss threshold used to construct the idealized trigger scenario would also generate more triggering events from which the statistical techniques could "learn", but since CAT bonds are typically constructed for relatively high return period losses (greater than 100 years), these solutions would not be relevant from a practical standpoint.

While accuracy is certainly an indispensable feature of any suitable technique for design of parametric trigger mechanisms, a technique should also produce trigger behavior that is meaningful from a physical perspective. Namely, a suitable technique for parametric trigger mechanism design should produce trigger behavior that reflects the seismic hazard and/or development patterns in the region of study. For this reason, the physical performance of the techniques' trigger mechanisms was evaluated representing earthquakes falling into the test set in maps. Figure B.2 shows the evaluation of two techniques that produce trigger mechanisms suitable from a numerical perspective: Neural Networks and the EA from Franco (2010). The plots include three layers: white points represent all earthquakes, and black points and gray triangles identify those in the test set with $B' = 0$ or $B' = 1$, respectively. While the plot on the left (Neural Networks) gathers all gray points in the center, the plot on the right (EA) shows more dispersion.

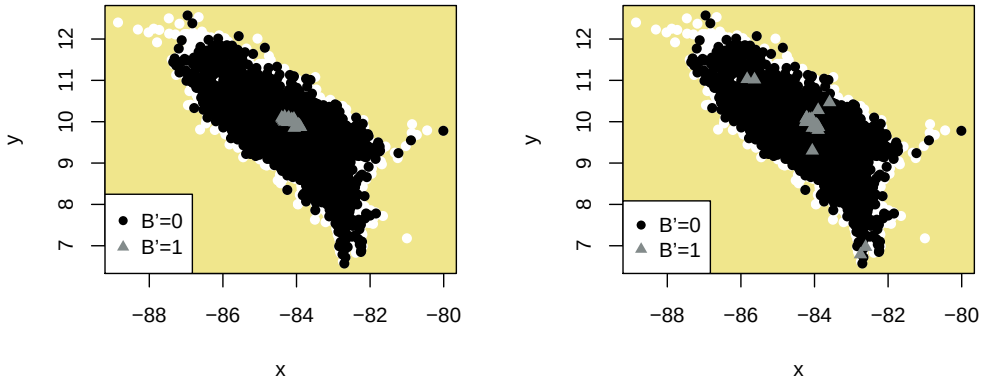


FIGURE B.2: Map of predictions obtained with Neural Networks (left) and the EA described in Franco (2010) (right).

5. Conclusions and Future Research

Natural catastrophes continue to cause enormous losses that remain largely uninsured, leaving populations vulnerable to severe financial impacts. The insurance and reinsurance industry, governments and catastrophe pools have started to employ financial instruments such as parametric CAT bonds to cede these catastrophic risks to the capital markets. Were these tools extended for more widespread usage at the retail level, we could progressively and massively reduce the "insurance gap" for earthquake risks. However, this requires the construction of accurate and unbiased parametric triggers with extreme efficiency and automation, something that is not available in the industry today.

To address this problem, we have explored solving the trigger design challenge as a classification problem, employing well-known and powerful techniques from statistics and machine learning. From a numerical perspective, it has been shown that these techniques can produce trigger mechanisms of equal or better accuracy than previously published techniques (Franco, 2010). And furthermore several statistical and machine learning methods provide huge efficiency gains in terms of decreasing classification time. Additionally, they provide scalability, being easily adapted to a larger parameter space and larger catalogs without losing much efficiency, and ease of implementation since there is a wide range of programs and programming languages that enable free and simple implementation of these statistical and machine learning techniques such as R (R Core Team, 2012), Octave (Eaton et al., 2014) and Scilab (Scilab Enterprises, 2012). Application of these statistical and machine learning techniques to the problem of parametric trigger design is not without complication, however, because while these methods provide accuracy and efficiency improvements, some of the examples shown in this paper produce trigger mechanisms with relatively low specificity values.

Several lines of future research emerge from the introduction of classification techniques to the development of trigger mechanisms for earthquake CAT bonds. First, it is apparent from the experiments in this paper that more meaningful insights as to the applicability of classification techniques to the development of trigger mechanisms could be gleaned from the use of a larger earthquake catalog. It would also be worthwhile to examine the behavior of the trigger mechanisms at multiple return periods, particularly lower ones. There is a natural imbalance in the data at high return periods since very few events trigger the bond. Consequently, there are two groups of events subjected to classification (depending on whether they should trigger a given CAT bond), but they greatly differ in size. Techniques may present low accuracy with respect to the minority (triggering) group and still have a good global accuracy. Analysis of the same simulated earthquake catalog at lower return periods would reduce this classification group imbalance but would not produce a usable trigger mechanism, since CAT bonds are typically constructed to protect against high return period losses. Therefore, such an experiment could provide valuable insights into the different classification techniques but would not produce directly usable trigger

mechanisms. A popular numerical alternative to this complication is to oversample events in the minority group, which would constitute an artificial expansion of the original earthquake catalog.

Introduction of such a large number of alternative techniques for parametric trigger mechanism design motivates the development of a selection framework. From the standpoint of practical implementation, it would be interesting to identify the most desirable characteristics for a trigger mechanism and order them. For instance, if accuracy is supreme, one should explore the use of more modern and complex techniques such as Random Forests and Multi-Layer Neural Networks (provided a larger catalog was available). In contrast, if the interpretability plays the largest role, it would make sense to employ more classical techniques and study graphical tools.

The technological developments characterizing the era of Big Data and the Internet of Things have potentially fascinating implications in this field. These avenues open the possibility of designing triggers not only based on few physical characteristics of an earthquake but on much more information obtained through broad networks of sensors. Metaheuristics, simheuristics (i.e., algorithms combining metaheuristics and simulation techniques) and other classical instruments may be used to perform a feature selection or extraction. Finally, the capacity of simulators to create larger catalogs is ever-increasing, constantly being able to generate more and more data, more and more reliably. In this scenario, non-linear approaches such as Deep Learning would be worth exploration.

Acknowledgments

This work has been partially supported by the Spanish Ministry of Economy and Competitiveness (grants TRA2013-48180-C3-P, DPI2013-44461-P, TIN2015-66951-C2-2-R), FEDER, the Department of Universities, Research & Information Society of the Catalan Government (Grant 2014-CTP-00001) and with the support of a doctoral grant from the UOC. The authors are also thankful to AIR Worldwide for permitting the usage of the AIR Earthquake Model for Central America version 1.1 for the reproduction of the example contained in Franco (2010).

References

Cardenas, V., S. Hochrainer, R. Mechler, G. Pflug, and J. Linnerooth-Bayer (2007). "Sovereign financial disaster risk management: The case of Mexico". In: *Environmental Hazards* 7, pp. 40–53.

Croson, D. C. and H. Kunreuther (1999). "Customizing reinsurance and CAT bonds for natural hazard risks". In: *Conference on Global Change and Catastrophic Risk Management*. Laxenburg, Austria.

Cummins, J. D. (2007). "CAT bonds and other risk-linked securities: state of the market and recent developments". In: *Social Science Research Network*. http://ssrn.com/abtract=105740.

Cummins, J. D. and M. A. Weiss (2009). "Convergence of insurance and financial markets: hybrid and securitized risk-transfer solutions". In: *Journal of Risk and Insurance* 76.3, pp. 493–545.

DeSarbo, W. S. and W. L. Cron (1988). "A maximum likelihood methodology for clusterwise linear regression". In: *Journal of classification* 5.2, pp. 249–282.

Eaton, J. W., D. Bateman, S. Hauberg, and R. Wehbring (2014). *GNU Octave version 3.8.1 manual: a high-level interactive language for numerical computations*. ISBN 1441413006. http://www.gnu.org/software/octave/doc/interpreter. CreateSpace Independent Publishing Platform.

Franco, G. (2010). "Minimization of trigger error in cat-in-a-box parametric earthquake catastrophe bonds with an application to Costa Rica". In: *Earthquake Spectra* 26.4, pp. 983–998.

— (2013). "Construction of customized payment tables for cat-in-a-box earthquake triggers as a basis risk reduction device". In: *Proceedings of the International Conference on Structural Safety and Reliability (ICOSSAR)*. New York, NY, pp. 5455–5462.

— (2014). "Earthquake mitigation strategies through insurance". In: ed. by M. Beer, I. A. Kougioumtzoglou, E. Patelli, and I. Siu-Kui Au. Springer Berlin Heidelberg. Chap. Encyclopedia of Earthquake Engineering", pp. 1–18.

Goda, K. (2013). "Basis risk for earthquake catastrophe bond trigger using scenario-based versus station intensity-based approaches: a case study for Southwestern British Columbia". In: *Earthquake Spectra* 29.3, pp. 757–775.

— (2014). "Seismic risk management of insurance portfolio using catastrophe bonds". In: *Computer-Aided Civil and Infrastructure Engineering* 30.7, pp. 570–582.

Grossi, P. and H. Kunreuther (2005). *Catastrophe modeling: a new approach to managing risk*. Springer.

Guy Carpenter (2014). *Chart: gap between economic and insured losses*. http://www.gccapitalideas.com/2014/01/20/chart-gap-between-economic-and-insured-losses/.

Hastie, T., R. Tibshirani, and J. Friedman (2009). *The elements of statistical learning*. 2nd ed. Springer.

Kotsiantis, S.B. (2007). "Supervised machine learning: A review of classification techniques". In: *Informatica* 31, pp. 249–268.

Lantz, B. (2013). *Machine learning with R*. Packt Publishing.

Pucciano, S., G. Franco, and P. Bazzurro (under review). "Loss predictive power for strong motion networks for usage in parametric risk transfer: Istanbul as a case study". In: *Earthquake Spectra*.

R Core Team (2012). *R: A language and environment for statistical computing*. ISBN 3-900051-07-0. http://www.R-project.org/. R Foundation for Statistical Computing. Vienna, Austria.

Scilab Enterprises (2012). *Scilab: Free and open source software for numerical computation*. http://www.scilab.org. Orsay, France.

Wald, D. and G. Franco (2017). "Financial decision-making based on near-real-time earthquake information". In: *Proceedings of the 16th World Conference on Earthquake Engineering*. paper: 3625. Santiago de Chile.

B.3 A simheuristic algorithm for the stochastic and capacitated multi-depot vehicle routing problem

Laura Calvet[1], Dandan Wang[2], Angel A. Juan[1]

1. Computer Science Department, IN3-Open University of Catalonia, Castelldefels, Spain
e-mail: {ajuanp, lcalvetl}@uoc.edu
2. GE (China) Research and Development Center Co., Ltd
e-mail: dandanw924@gmail.com

Abstract

When capacity constraints exist at each depot, the Multi-Depot Vehicle Routing Problem (MDVRP) is a non-trivial extension of the well-known Capacitated Vehicle Routing Problem (CVRP), since it combines a customer-to-depot assignment problem with several CVRPs. In real-life scenarios, it is usual to find uncertainty in the customers' demands. There are few works on the Stochastic MDVRP and, to the best of our knowledge, all assume unlimited capacity at each depot. This paper presents a simheuristic framework combining Monte Carlo simulation with a metaheuristic algorithm to deal with this problem. Its efficiency is tested on a set of stochastic instances.

Keywords: logistics & transportation, multi-depot vehicle routing problem, stochastic optimization problems, simheuristics.

1. Introduction

A considerable number of decision-making problems in fields such as logistics and transportation, finance, or production, can be formulated as Combinatorial Optimization Problems (COPs). In these problems, the goal is to find, in a reasonable amount of time, an optimal or near-optimal solution from a finite —although usually very large— set of possible combinatorial alternatives. Design of distribution routes, portfolio management, facilities location, and flight scheduling, are some of the many popular examples of COPs. In the scientific literature, it has been frequently assumed that these problems were deterministic in nature, i.e., that all the required information was available in advance. Although this assumption makes optimization easier, in most cases is quite unrealistic and may lead to solutions of poor quality for real-life scenarios characterized by high levels of uncertainty. Therefore, there is a need for developing new methods able to provide robust and risk-aware solutions to COPs under uncertain and dynamic environments.

According to Bianchi et al. (2009), COPs may be classified depending on the way uncertain information is formalized. In Stochastic COPs (SCOPs), uncertainty is identified with random variables following a probability distribution (Juan et al., 2011b). Fuzzy COPs deal with fuzzy quantities and constraints with fuzzy sets (Erbao and Mingyong, 2009). Robust COPs analyze information given in the form of interval values (Kouvelis and Yu, 1997). In all these cases, problems may be solved by designing a solution usable for any specific scenario (aprioristic or robust optimization) or by taking decisions each time uncertain information is revealed (online or reactive optimization). When there is complete uncertainty, the last strategy is mostly preferred (Jaillet and Wagner, 2008). Aprioristic optimization may lead to a poor performance as it is often difficult or unfeasible to find a solution that provides satisfactory results in all possible scenarios. On the other hand, applying online optimization may be time-consuming and requires having computing resources permanently available. When there is a set of similar problem instances to solve, it can be advantageous employing always the same (or a similar) solution. For example, a company that each day sends goods to retailers will probably prefer to design specific routes and always use them, because drivers will perfectly know them, and customers will appreciate to get to know their driver and receive the goods each day at the same time. An intermediate approach consists in computing an aprioristic solution and, as events occur, allowing the possibility of taking corrective actions, i.e., adapting the solution each time new information is revealed.

Some of the earliest works in the field of Stochastic Vehicle Routing Problems were based on exact methods (Laporte et al., 1994; Gendreau et al., 1995), which guarantee the optimality of the solution. However, due to the complexity of these problems, those approaches are only feasible

for small-scale instances. In contrast, approaches including heuristics (Dror and Trudeau, 1986) and metaheuristics (Bianchi et al., 2006; Moghaddam et al., 2012) usually provide near-optimal solutions in reasonable times, even for realistic size instances. This paper addresses the Stochastic version of the Capacitated Multi-Depot Vehicle Routing Problem (or Stochastic and Capacitated MDVRP), where customers' demands are assumed to be random variables –a similar approach to the one introduced here could be applied if the transportation times were random variables too. The MDVRP with capacity constraints is a challenging extension of the Capacitated Vehicle Routing Problem (CVRP), which combines a customer-to-depot assignment problem with several CVRPs (Figure B.1).

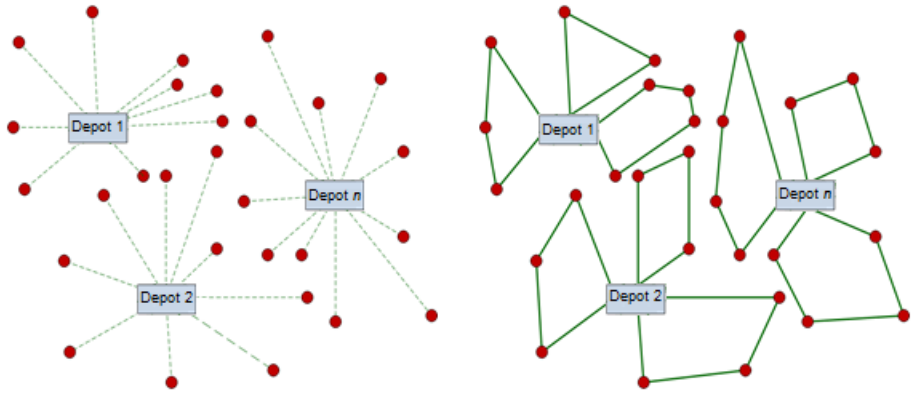


FIGURE B.1: Customer-depot assignment (left) and posterior routing (right) processes.

Since assignation and routing issues are often interrelated, this problem is a two-stage decision process, where the assignment map will affect the quality of the posterior routing. Montoya-Torres et al. (2015) highlight a noticeable growing interest in the MDVRP during the last decade, with over 103 publications between 2006 and 2014. In the Stochastic and Capacitated MDVRP, a set of customers with random demands must be served by a fleet of homogeneous capacitated vehicles departing from one among several capacitated depots. The main goal of this problem is to determine the set of routes that minimizes the expected total routing cost –including recursive actions–, subject to a number of capacity-related constraints. The problem has numerous applications in real-life, e.g.: garbage collection, gas distribution, stocking of vending machines, and other similar activities in which the specific amount of goods to leave or pick up is not known until the place is reached. Despite its relevance, there are few works on the Stochastic MDVRP. This paper aims to reduce the gap between theory and practice by introducing a simheuristic algorithm (Juan et al., 2015a). Simheuristics is an optimization-simulation framework that combines metaheuristics (in a general sense, i.e., including heuristics, matheuristics, and hyperheuristics) and simulation (Monte Carlo, discrete-event, agent-based, etc.) to solve SCOPs. This hybrid approach has been successfully implemented in several fields, including: Internet computing (Cabrera et al., 2014), transportation and logistics (Juan et al., 2011b; Juan et al., 2014b; Gonzalez et al., 2016), and production scheduling (Juan et al., 2014a). The main advantages of simheuristics are their flexibility (since they only assume that each random variable follows a known probability distribution with an existing mean), accuracy, and relatively ease of implementation. Accordingly, the main contributions of this work are: *(i)* to the best of our knowledge, this is the first paper that addresses the Stochastic and Capacitated MDVRP, which is a realistic extension of the CVRP; *(ii)* it proposes a novel simheuristic framework that combines different metaheuristics –a Multi-Start (MS) (Martí et al., 2013), an Iterated Local Search (ILS) (Lourenço et al., 2010), and a Large Neighborhood Search (LNS) (Pisinger and Ropke, 2010)–, with Monte Carlo simulation (MCS); and *(iii)* it proposes a well-defined set of benchmarks (i.e., reproducible) that extend, in a natural way, the ones traditionally employed in the deterministic version of the Capacitated MDVRP.

The remaining of this paper is structured as follows. Section 2 provides a description of the Stochastic and Capacitated MDVRP. Section 3 reviews the related work. The proposed approach and its implementation are presented in Section 4. Section 5 describes the computational experiments carried out to illustrate and test our algorithm. Section 6 analyses the results obtained in

these experiments. Finally, Section 7 summarizes the main conclusions of this paper and identifies some lines of research for future work.

2. The Stochastic and Capacitated MDVRP

The Stochastic and Capacitated MDVRP is characterized by the randomness of at least one of its parameters or structural variables. These random variables follow specific probability distributions. This problem may be seen as a non-trivial extension of the Stochastic CVRP (Gendreau et al., 1996; Stewart and Golden, 1983). There are three problems belonging to this family: the CVRP with Stochastic Demands, which is the most popular (Bianchi et al., 2006); the CVRP with Stochastic Customers (Bertsimas, 1988); and the CVRP with Stochastic Times (Laporte et al., 1992; Kenyon and Morton, 2003). The Stochastic and Capacitated MDVRP is a SCOP that may be described as follows. Let $G = \{V, E\}$ be a complete directed graph, where $V = \{V_d, V_c\}$ is the set of vertices including the depots (V_d) and the customers (V_c), and E is the set of edges connecting all vertices in V. Each customer $i \in V_c$ has a positive demand D_i that follows a probability distribution, either theoretical or empirical. It is assumed that its mean, denoted by $E[D_i]$, exists. While this distribution is known beforehand, the exact demand cannot be revealed until the vehicle reaches the customer. Each depot $p \in V_d$ has assigned a maximum number of vehicles, m. All vehicles are supposed to have the same capacity W, which is greater than the highest demand. Each edge in E has an associated cost $c_{ij} = c_{ji} \geq 0$, that usually depends on the distance between vertices i and j. A solution of this problem is a set of routes in which each route starts at one depot in V_d, connects one or more customers in V_c, and ends at the same depot (Figure B.1). Moreover, each customer must be visited only once, except in the undesirable case in which a route failure occurs.

The classical goal here is to find a feasible solution that minimizes the expected routing cost while satisfying the customer demands, and the constraints related to the number of vehicles, and the vehicles' capacity. However, other constraints may also apply, e.g.: a maximum allowable cost for a route, time windows for visiting each customer, etc. In addition, different goals may be proposed such as solution balance or minimization of environmental costs. Even in its simplest version, this problem represents a challenge since it integrates a combinatorial assignment problem –in which each customer is assigned to one depot– with several Stochastic CVRPs, one per depot. The additional complexity lies in the interrelation between assigning and routing issues.

One way to model a Stochastic and Capacitated MDVRP is as a two-stage problem. In the first stage (design stage), a set of routes is designed considering the probability distributions associated with each customer's demand. The second stage (corrective stage) specifies the actual route of each vehicle, which may include corrective actions if the route fails, i.e., if the demand of a customer visited by a given vehicle is higher than the remaining vehicle capacity. In this case, the vehicle must return to the depot to reload. Often, the possibility of re-stocking is allowed, that means that a vehicle may return to the depot before it has run out of capacity. For instance, if the remaining vehicle capacity is not enough to satisfy the expected demands of the customers that still have to be served. The solution must minimize the expected total cost, which is the sum of the costs of the routes planned in the first stage (fixed cost), and the expected costs due to corrective actions (variable cost).

3. Related Work

Despite the growing interest in the MDVRP and in analyzing COPs under realistic assumptions, including uncertainty (Caceres et al., 2014), there are just a few works focused on the Stochastic MDVRP. For this reason, this section will first review the literature on the (deterministic) MDVRP and some extensions. Then, a number of relevant works addressing the Stochastic VRP and other related problems will be also reviewed. Finally, the section will conclude with some comments regarding the state-of-the-art in the Stochastic MDVRP.

The Multi-Depot Vehicle Routing Problem and extensions

The MDVRP has been intensively studied in the last decades. The first works were published in the late 1960s and 1970s, and relied on heuristics or exact methods. Metaheuristics became

more popular during the following decades. More recently, powerful hybrid algorithms are being proposed. Table B.1 presents related works.

The VRP with Stochastic Demands and related problems

Regarding the VRP with Stochastic Demands (VRPSD), the first works appear in the 80s. Table B.2 shows some related works. It is worth highlighting a few outstanding contributions. In Dror and Trudeau (1986), the concept of route failure is introduced, and its effects on the expected cost of a route are illustrated. A review on the Stochastic CVRP is presented in Gendreau et al. (1996), where the main variants are presented. Yang et al. (2000) suggest anticipating possible stock-outs by incorporating preventive breaks or restocking in the route design. The aim is to reduce the probability of route failure and, as a result, the cost. During the last decades, the scientific community has focused on the implementation of metaheuristics. In this context, Bianchi et al. (2006) compare the performance of several methodologies embedding one of the following metaheuristics: SA, TS, ILS, Ant Colony Optimization, and Evolutionary Algorithm (EA).

There are several simheuristic-based approaches that tackle routing SCOPs. Since we propose a simheuristic algorithm, they are presented in more detail. Juan et al. (2011b) present a procedure following these steps: *(i)* transform a VRPSD instance into a set of different CVRP instances by employing the mean values of the demand distributions as actual demands and using different safety stock levels in the vehicles; *(ii)* solve each of the CVRP instances (one per level) with an efficient metaheuristic; and *(iii)* estimate the reliability of the CVRP solution for the VRPSD and the cost of potential corrective actions by employing MCS. An enhanced version of the algorithm, including parallelization issues, is presented in Juan et al. (2013b). Finally, Gonzalez et al. (2012) address the Arc Routing Problem (ARP) with Stochastic Demands. The procedure also relies on MCS and an existing metaheuristic for solving the Capacitated ARP.

The Stochastic Multi-Depot Vehicle Routing Problem

The number of works analyzing the Stochastic MDVRP is rather limited. Tillman (1969) expands the CWS heuristic to address it. The procedure proposed may be applied to demands with Poisson, Exponential, Normal, Binomial or Chi-squared distributions. In Chan et al. (2001), a Multi-Depot, Multiple-Vehicle, Location Routing Problem with Stochastically Processed Demands is formulated. The probable demands are estimated by stochastic processes before the vehicle location-routing decisions. Tatarakis and Minis (2009) study the Stochastic MDVRP considering both the case in which products are stored dedicatedly or together in a compartment. Dynamic programming algorithms are proposed to determine the minimal routing cost, and an optimal routing policy is derived to decide whether a vehicle has to return to the depot for a reload after serving the current customer or should continue to the next customer. Tauhid et al. (2012) solve the Stochastic MDVRP in three phases: first a Nearest Neighbor classification method is used for grouping the customers; then, the sum-of-subsets method is applied for routing; and finally, the routes are optimized throughout a greedy method. They aim to minimize the number of routes and, accordingly, the number of vehicles needed.

None of the aforementioned works deals with the Stochastic and Capacitated MDVRP analyzed here. In particular, Tillman (1969) and Tauhid et al. (2012) consider unlimited capacities at each depot -which significantly reduces the difficulty of the problem and constitutes an unrealistic assumption. In addition, Tauhid et al. (2012) do not really consider stochastic demands. Also, Tillman (1969) makes strong assumptions on the probability distributions of these demands. Chan et al. (2001) and Tatarakis and Minis (2009) deal with problems that, although somewhat related, can not be considered Stochastic and Capacitated MDVRPs. While the former focuses on location issues, in the latter a single vehicle must deliver multiple products given a predefined customer sequence.

4. A Simheuristic for the Stochastic and Capacitated MDVRP

General overview

Our approach relies on two facts: *(i)* the Stochastic and Capacitated MDVRP can be considered a generalization of the Capacitated MDVRP, i.e., the Capacitated MDVRP can be seen as a Stochastic and Capacitated MDVRP in which the random demands have zero variance; and *(ii)* despite

TABLE B.1: Works on the MDVRP and extensions

Work	Problem	Approach
Tillman and Cain (1972)	MDVRP	Branch and bound methods
Golden et al. (1977)	Travelling Salesman Problem (TSP), Multiple TSP, CVRP and MDVRP	Heuristics
Raft (1982)	MDVRP	Heuristic-based technique, which is decomposed into: route assignment, depot assignment, vehicle assignment, delivery period, and route design
Renaud et al. (1996)	Capacitated MDVRP with a Maximum Route Length	Tabu Search (TS)
Cordeau et al. (1997)	Periodic VRP, Periodic TSP and Capacitated MDVRP	TS
Salhi and Sari (1997)	MDVRP determining the Vehicle Fleet Composition	Three-level methodology, including composite heuristics
Thangiah and Salhi (2001)	MDVRP	Genetic Algorithm (GA), an insertion heuristic and a post-optimization method
Wu et al. (2002)	Capacitated Multi-Depot Location-Routing Problem with a Heterogeneous Fleet and a Limited Number of Available Vehicles	Simulated Annealing (SA)
Bae et al. (2007)	Capacitated MDVRP	GA and GUI-type programming
Crevier et al. (2007)	MDVRP with Inter-Depot Routes	TS, a solution pool, a route generation algorithm, a set partitioning algorithm, and a post-optimization process
Pisinger and Ropke (2007)	CVRP, CVRP with Time Windows, Capacitated MDVRP, Site-Dependent CVRP, and Open CVRP	LNS
Chen and Xu (2008)	Capacitated MDVRP	GA and Metropolis acceptance rule of the SA
Ho et al. (2008)	MDVRP	Two hybrid GAs
Dondo and Cerdá (2009)	MDVRP with Time Windows	Mixed-integer linear programming formulations, a spatial decomposition scheme and a local search improvement algorithm exploring large neighborhoods
Mirabi et al. (2010)	Capacitated MDVRP	Three hybrid heuristics including Nearest depot method, Clarke and Wright Savings (CWS) heuristic (Clarke and Wright, 1964), Nearest Neighbor heuristic, and SA
Gulczynski et al. (2011)	Multi-Depot Split Delivery VRP	Integer programming-based heuristic
Surekha and Sumathi (2011)	Capacitated MDVRP	Clustering, CWS heuristic, and a GA
Cordeau and Maischberger (2012)	Capacitated MDVRP among other routing problems	ILS and TS
Vidal et al. (2012)	Capacitated MDVRP, Periodic VRP, and Capacitated Multi-Depot Periodic VRP	Hybrid GA
Juan et al. (2015c)	Capacitated MDVRP	ILS and biased randomization
Escobar et al. (2014)	Capacitated MDVRP	Hybrid Granular TS
Karakatic and Podgorelec (2015)	MDVRP	Survey of GAs
Li et al. (2015)	Capacitated MDVRP with Simultaneous Deliveries and Pickups	ILS

TABLE B.2: Works on the VRPSD and related problems

Work	Problem	Approach	Modeling
Stewart and Golden (1983)	VRPSD	Formulations and heuristic algorithms	Demands follow Normal or Poisson distributions
Dror and Trudeau (1986)	VRPSD	Modification of the CWS heuristic	Demands follow specific distributions from a limited set
Bastian and Kan (1992)	VRSPD	Formulation for a penalty model, a chance-constrained model, and a full-service model	Demands are independent and identically distributed with known probability distributions
Gendreau et al. (1995)	VRPSD with Stochastic Customers	Stochastic integer formulation and integer L-shaped method	Demands are independent, discrete and identically distributed with known probability distributions
Yang et al. (2000)	VRPSD	Two heuristics	Demands are independent and identically distributed with known probability distributions
Tan et al. (2007)	VRPSD with Time Windows	Multi-objective EA and simulation techniques	Demands follow Normal distributions
Ismail and Irhamah (2008)	VRPSD	GA and TS	Demands follow independent and discrete Uniform distributions
Moghaddam et al. (2012)	CVRP with Uncertain Demands	Particle Swarm Optimization	No assumptions are made
Goodson (2015)	Multi-Compartment VRPSD	Method to calculate the expected cost of a solution integrated in a cyclic-order-based SA	Demands follow discrete distributions

the fact that the Stochastic and Capacitated MDVRP has not been intensively studied, there exists efficient algorithms for solving the Capacitated MDVRP.

The general ideas behind our approach are described next. Initially, given an instance of the Stochastic and Capacitated MDVRP, it is transformed into a deterministic instance by replacing each random variable by its mean. A set of high-quality solutions for the deterministic version is then obtained by using an efficient algorithm. While the search takes place, MCS techniques are employed to assess the performance of these promising solutions for the stochastic version. We define the best solution as the one with the lowest expected total cost. This assessment is carried out according to the following steps: *(i)* run hundreds of executions (or thousands, if more time is available), where each execution implies the generation of random values for each random demand according to the associated probability distribution; *(ii)* assess the performance of each solution by estimating the expected total cost as the average of the total costs obtained at each execution; and *(iii)* use the simulation feedback to better guide the searching process inside the metaheuristic. In this step it is assumed that there is a correlation between the solutions for the Capacitated MDVRP and those for the Stochastic and Capacitated MDVRP. In other words, the best solutions for the deterministic version are likely to be also high-quality solutions for the stochastic version. The correlation is expected to be stronger (or more significant) as the demand variances tend to zero.

Our methodology employs safety stocks as suggested in Juan et al. (2011b). A safety stock is a certain amount of the vehicle capacity that is not considered while designing the routes. Then, if the final routes' demands surpass their expected values, this stock can be employed to try to satisfy them. Thus, the aim of considering safety stocks is to reduce the probability of a route failure.

Proposed steps

The flowchart diagram of our approach is depicted in Figure B.2 and described next (additional low-level details are provided in the next subsection):

1. Consider a Stochastic and Capacitated MDVRP instance defined by a set of n customers. Each customer i has associated a demand D_i ($1 \leq i \leq n$) that follows a known probability distribution with an existing mean $E[D_i]$.

2. Determine a set K of percentages, where each element k_l is the percentage of the vehicle capacity (W) that can be used during the route design phase; in other words, $1-k_l$ represents a fixed level of safety stock. For each of these elements, follow the steps 3 to 9.

3. Consider the Capacitated MDVRP(k_l) with a total vehicle capacity of $W_l^* = k_l \cdot W$ and deterministic demands $d_i = E[D_i]$.

4. Generate an initial solution for the Capacitated MDVRP(k_l). This solution is also an aprioristic solution for the Stochastic and Capacitated MDVRP. It will be employed "as it is" as long as there is no need of corrective actions (routes failures and re-stockings). Therefore, the cost associated to this solution, $C_{C.MDVRP}(k_l)$, can be considered a base or fixed cost of the Stochastic and Capacitated MDVRP solution. In the case of the stochastic problem, there is also a variable cost $C_{CA}(k_l)$ that depends on the corrective actions undertaken. Consequently, for a given value of k_l, the total cost of the 'stochastic' solution (the one associated with the Stochastic and Capacitated MDVRP) is the sum of the fixed cost corresponding to the 'deterministic' solution (the one associated with the Capacitated MDVRP) and the variable cost due to corrective actions, $C_{S.C.MDVRP}(k_l) = C_{C.MDVRP}(k_l) + C_{CA}(k_l)$.

5. Use MCS to estimate the expected cost due to corrective actions for each route j of the aprioristic solution, $E[C_{CA}^j(k_l)]$ ($1 \leq j \leq m$). Then, aggregate the expected total cost for all routes, $E[C_{CA}(k_l)]) = \sum_{j=1}^{m} E[C_{CA}^j(k_l)]$. In this phase, a short simulation (i.e., one with a 'reduced' number of runs) is used to quickly get that estimate. Then, the expected total cost of the solution is calculated as follows: $E[C_{S.C.MDVRP}(k_l)] = C_{C.MDVRP}(k_l) + E[C_{CA}(k_l)]$.

6. Set a base solution as the initial solution.

7. Employ a metaheuristic algorithm, which starts an improvement process that will continue until a stopping condition, based on time or a fixed number of iterations, is reached. At each iteration the following steps are implemented. First, a perturbation is applied to the base

solution to generate a new one. If the fixed cost of the new solution is lower than the fixed cost of the current base solution, then the list of the best deterministic solutions is updated (only if it is not full or if the worst solution has a higher cost, then a swap is performed) and the expected total cost of the new solution is estimated with a short simulation. If this cost is lower than the expected total cost of the base solution, the latter is replaced and the list of the best stochastic solutions is updated. Otherwise, an acceptance criterion is used to decide whether the base solution is deteriorated to the new one. Before that, if the fixed cost of the new solution is higher, then that solution is discarded. This iterative process will provide, after analyzing many possible solutions, a list of promising solutions for the Stochastic version of the Capacitated MDVRP.

8. Try to improve all promising solutions with an intensive routing algorithm.
9. Use a long simulation (i.e., one with a 'large' number of runs) to generate a sample of total costs for each promising solution. Large samples are required to obtain estimates with small confidence intervals.
10. Finally, return the top best stochastic solutions (considering all solutions found with the different values in K), and the corresponding samples (they will be used for completing a risk analysis).

Details and further considerations

Some key issues of our approach should be explained in detail. The algorithm used to get the initial solution (step 4) for the Capacitated MDVRP is the one proposed in Juan et al. (2015c) and has two stages. Initially, a customer-depot assignment map is set, and then the savings heuristic (Clarke and Wright, 1964) is applied to obtain a fast routing plan. The first step is performed by computing a priority list of potentially eligible customers for each depot. These lists are sorted according to a distance-based criterion called marginal savings: for a given depot $r \in V_d$ and a customer $i \in V_c$, the marginal saving is calculated as the difference between the distance-based cost of assigning i to the closest depot q ($q \neq r$), and the distance-based cost of assigning i to r. Afterwards, each list is randomized through a Geometric distribution as described in Juan et al. (2010). This technique makes it possible the generation of numerous lists without losing the logic behind the priority list based on the computed marginal savings. Then, two policies are iteratively considered to assign customers to depots: the first allows the depot with the most remaining serving capacity to choose the next customer from its priority list, while the second employs a round robin criterion to select which depot chooses next. The second step consists in solving each resulting CVRP independently by employing the savings heuristic.

Regarding safety stocks, it is expected that lower values of k_l will provide more reliable routes (i.e., routes with a lower probability of failure), as a high percentage of the vehicle capacity will be reserved as safety stock. However, a high fixed cost will result too, since more vehicles will be needed to cover all the customers' demands. In the worst case, the problem instance could become unsolvable. On the other hand, a high value of k_l is related to a lower fixed cost but a higher variable cost due to the elevated risk of having to return to the depot to reload. Considering the trade-off between these two costs, we try different values as indicated in step 2.

The cost due to corrective actions (step 5) is computed as follows. In case of route failure, it includes the cost of returning to the depot first and then to the customer being served. It is assumed that the vehicle delivers all the remaining stock before going back to reload. A re-stocking is carried out when the expected demand of the next customer is higher than the current remaining stock. The cost of this strategy incorporates the costs on the edges that link a customer with the depot and the depot with the next customer minus the cost of the edge linking both customers. Other re-stocking policies could be tested.

The perturbation operator (step 7) employed in the iterative part of the metaheuristic framework modifies the current solution by reallocating a given percentage p of customers, considering the remaining capacity of the depots, and the distance-based cost for each pair of customer and depot. The savings heuristic is applied again to design the routes. A Demon-like acceptance criterion (Talbi, 2009) is used to diversify the search. It allows the base solution to be deteriorated to a new solution if two conditions are met: *(i)* no consecutive deteriorations take place; and *(ii)*

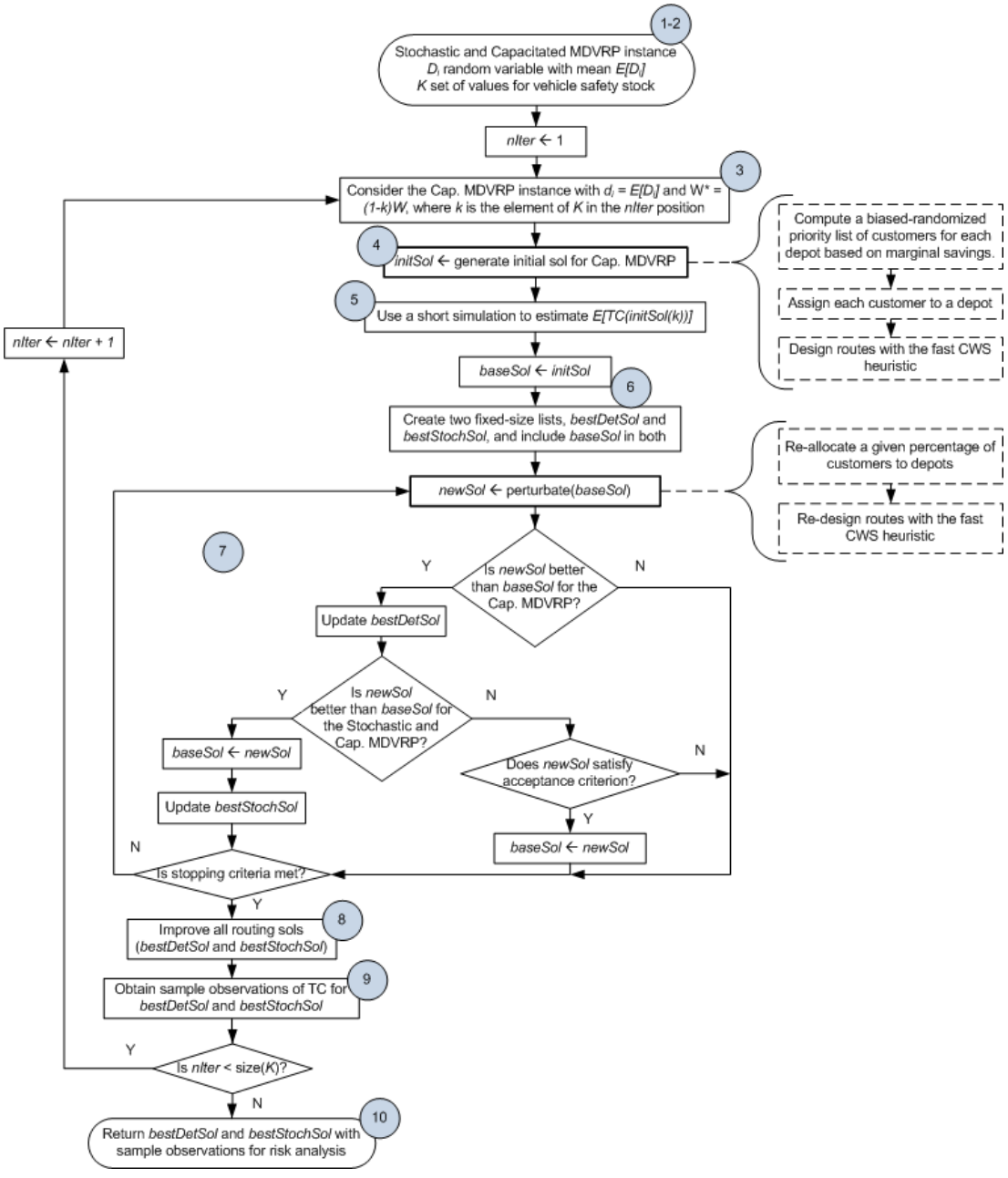


FIGURE B.2: Flowchart diagram of the proposed approach

TABLE B.3: Description of the benchmark instances

Instance	N. customers	N. vehicles	N. depots	Max. route length	V. max. cap.
p01	50	4	4	n/a	80
p02	50	2	4	n/a	160
p03	75	3	5	n/a	140
p04	100	8	2	n/a	100
p05	100	5	2	n/a	200
p06	100	6	3	n/a	100
p07	100	4	4	n/a	100
p08	249	14	2	310	500
p09	249	12	3	310	500
p10	249	8	4	310	500
p11	249	6	5	310	500
p12	80	5	2	n/a	60
p13	80	5	2	200	60
p14	80	5	2	180	60
p15	160	5	4	n/a	60
p16	160	5	4	200	60
p17	160	5	4	180	60
p18	240	5	6	n/a	60
p19	240	5	6	200	60
p20	240	5	6	180	60
p21	360	5	9	n/a	60
p22	360	5	9	200	60
p23	360	5	9	180	60

the degradation does not exceed the value of the last improvement. The reason to implement this acceptance criterion is to reduce the risk of getting trapped in a local optimum.

In order to improve the most promising solutions found within the metaheuristic framework, the routing algorithm proposed by Juan et al. (2011a) is applied to each one. This algorithm is based on a randomized version of the savings heuristic that employs a Geometric distribution to guide the random search, and a cache and splitting techniques to make it more efficient. This algorithm has been adapted for the stochastic solutions.

Finally, it is interesting to observe that, considering the parameters (not the estimates), the fixed cost and the expected total cost of the best deterministic solution represent a lower and an upper bound, respectively, of the expected total cost associated to the best stochastic solution. The set of samples will allow us to compare the solutions not only focusing on the expected total cost, but also on the distribution of the total cost.

5. Computational Experiments

The algorithm described in the previous section has been implemented as a Java application. A standard personal computer, Intel QuadCore i5 CPU at 3.2 GHz and 4 GB RAM with Windows 7, has been used to execute all tests. Our algorithm was tested on 23 MDVRP benchmark instances: the first seven instances were proposed by Christofides and Eilon (1969), the following four were created by Gillett and Johnson (1976) and the remaining are described in Chao et al. (1993). Vidal et al. (2012), Escobar et al. (2014), and Juan et al. (2015c) are some recent works using them. The best known solutions have been extracted from these works. Table B.3 summarizes the main instance characteristics. It includes the instance name, the number of customers, the maximum number of vehicles per depot, the number of depots, the maximum route length allowed, and vehicles maximum capacity. These instances have been adapted for the Stochastic and Capacitated MDVRP as described next. The demand of each customer (d_i) has been considered as a random variable D_i following a Lognormal distribution with mean d_i and variance vd_i. Three different scenarios have been considered, each one with a respectively different variance: 0.1 $E[D_i]$, 0.5 $E[D_i]$, and 1 $E[D_i]$, where $E[\cdot]$ represents the mean or expected value (Figure 3). In order to choose the percentage of the vehicle capacity in the route design phase ($1 - k_l$), 5 equally-spaced values varying from 0.90 to 1.00 have been tested.

TABLE B.4: Parameters' values for each algorithm

	ILS	LNS	MS
bM	[0.3, 0.4]	[0.2, 0.3]	[0.3, 0.4]
bR	[0.2, 0.3]	[0.2, 0.3]	[0.1, 0.2]
p	[0.3, 0.4]	[0.1 – 0.4]	[0.2, 0.3]
iT	N/A	100	N/A
α	N/A	0.97	N/A

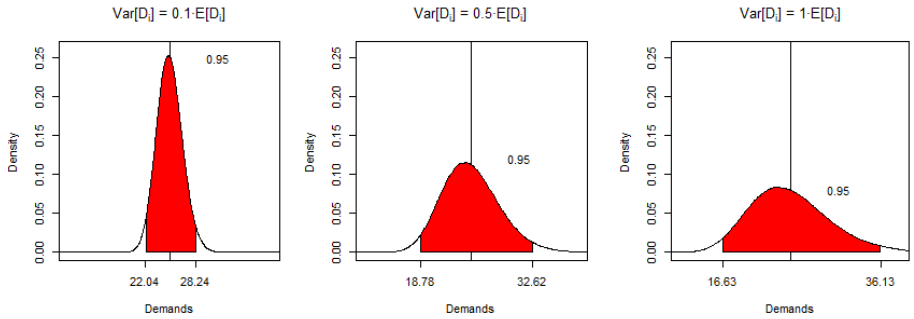


FIGURE B.3: Probability distributions with different levels of demand variability, for $E[D_i] = 25$

In addition to test the simheuristic algorithm described, a computational experiment is performed where other metaheuristics are considered. This allows us to compare them in terms of performance. In particular, there are 2 more algorithms based on:

- a Multi-Start algorithm: the base solution (and, as a consequence, the perturbation procedure) is erased, so a new solution is created in each iteration of the loop.
- a Large Neighborhood Search algorithm: While in the ILS algorithm a Demon-based acceptance criterion was implemented, here one based on a Simulated Annealing (Nikolaev and Jacobson, 2010) is applied. It requires an initial temperature (iT) and a parameter (α) which controls the temperature's update.

The computational time is limited to 30 seconds, which seems a reasonable time period for real-life applications. The number of seeds is set to 10, and only the best result are stored (notice that these runs can be executed in parallel, and thus a decision maker is mainly interested in the best solution). Concerning the number of iterations in each simulation, we have employed 200 runs (observations) for the short simulations (this allows us to obtain rough estimates of the stochastic costs in a reasonable amount of computing time) and 2,000 runs for the long simulations (which allows us to obtain more accurate estimates of the stochastic costs for each of the promising solutions). The selection of these values, as well as of the number of solutions stored in the list of top solutions (4 in our experiments), is mainly driven by the total computing time available. If more time is available, then these values can be incremented in order to obtain even better and/or more accurate results. Biased randomization techniques are used in the generation and repair of solutions, and in the intensive routing algorithm. These techniques rely on two Geometric distributions (one for mapping and one for routing) and, therefore, they require distribution parameters: bM and bR, respectively. Additionally, there is a parameter p which controls the percentage of nodes that may be reallocated in a solution when perturbing it. All these parameters are tuned independently for each algorithm (ILS, LNS, and MS) following these steps: *(i)* randomly select three instances; *(ii)* design a full factorial experiment, considering that each parameter can be randomly chosen from a 'reasonable' range; and *(iii)* select the values providing the best results. In the case of the LNS-based approach, iT and α are tuned following the same steps, and p is initially set at a specific value which is increased until reaching a maximum, adding the same quantity at each iteration of the loop. Values selected are shown in Table B.4.

Results are displayed in Tables B.5, B.6, and B.7. Each of these tables represents a specific scenario. The first column identifies the instance and the second shows the *best-known solution* (BKS) for the deterministic Capacitated MDVRP. The next six columns are associated with the

solution with the lowest fixed cost: the first, the *best deterministic solution - fixed cost* (BDS-FC), represents the fixed cost; the second calculates the gap between the BKS and the BDS-FC, which reveals the performance of our algorithm for the deterministic version of the problem; the third, the *best deterministic solution - total expected cost* (BDS-TEC), is the expected total cost; the fourth, the *best deterministic solution - reliability* (BDS-R), has been computed as one minus the number of route failures divided by the number of routes; the fifth, *best deterministic solution - k* (BDS-K), provides the percentage of the vehicle total capacity chosen; and the sixth, *best deterministic solution - time* (BDS-T), indicates the seconds that the execution has lasted. The following column represents the gap between the BDS-TEC and the BDS-FC. The next four columns are associated with the solution with the lowest expected total cost: the *best stochastic solution - total expected cost* (BSS-TEC) contains the expected total cost; the following three columns, the *best stochastic solution - reliability* (BSS-R), the *best stochastic solution - k* (BSS-K), and the *best stochastic solution - time* (BSS-T), show the associated reliability, *k*-value, and time required, respectively. The next two columns are the gaps between the BSS-TEC and the BKS, and between the BSS-TEC and the BDS-FC, respectively. It is important to highlight that, provided a 'large' number of simulation iterations is used, the BSS-TEC is bounded by the BDS-FC and the BKS (lower bounds), and the BDS-TEC (upper bound). Therefore, the previous gaps show the difference between the BSS-TEC and its lower bounds. The last column is the gap between the expected total costs of both solutions.

6. Analysis of results

The results obtained by the ILS-based simheuristic algorithm show that assuming a problem being deterministic can lead to solutions with poor performance even in scenarios characterized by demands with a relatively low variance. In all experiments, the expected total cost obtained with the best stochastic solution is better than the one obtained with the best deterministic solution. There is a case in which the gap reaches the −9.39% (instance p08 with high variance). The reason is that the deterministic solution is not balanced, and a high variance results in an increasing of the expected total cost. Figure B.4 illustrates the case of the instance p02 with a high variance. The vehicle capacity is 160. The left and right plots represent the best deterministic solution and the best stochastic solution, respectively. The numbers in the nodes reveal the expected customer demands, while the numbers in the center of each route are the total demands. Although the routes are similar, notice that the best stochastic solution seems more 'balanced' in terms of demands, which explains why it is also more reliable.

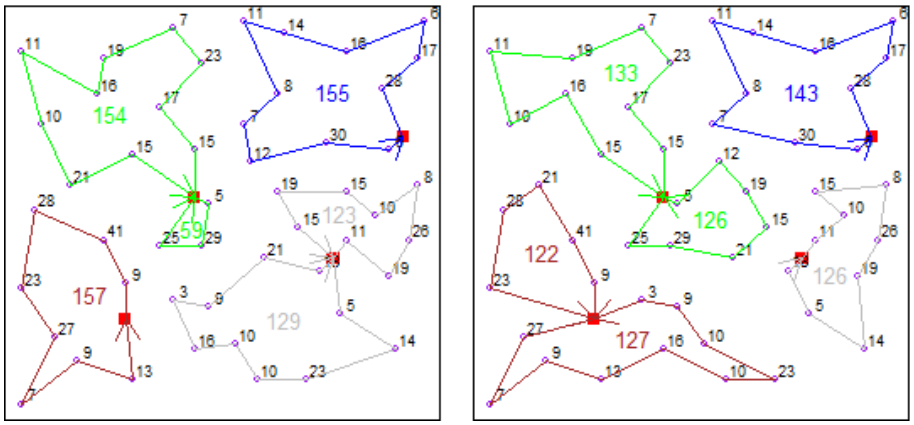


FIGURE B.4: Best deterministic (left) and stochastic solutions (right) for instance p02

Table B.8 summarizes the results described in Tables B.5, B.6, B.7. For each scenario, it shows the mean gaps. The gaps between the BDS-FC and the BKS, which ranges from 1.74% to 1.83%, show that our approach is relatively competitive for finding the best solution to the deterministic problem. The third column reveals that the difference between the BDS-TEC and the BDS-FC (i.e., the total cost if there was no stochasticity) is positive and positively correlated with the

TABLE B.5: Costs considering different instances with a low demand variability

Inst.	BKS (1)	BDS-FC (2)	G. (2)-(1)	BDS-TEC (3)	BDS-R	BDS-K	BDS-T	G. (3)-(2)	BSS-TEC (4)	BSS-R	BSS-K	BSS-T	G. (4)-(1)	G. (4)-(2)	G. (4)-(3)
p01	576.87	583.11	1.08%	594.85	1.00	1.00	0	2.01%	594.85	1.00	1.00	0	3.12%	2.01%	0.00%
p02	473.53	480.72	1.52%	487.43	1.00	1.00	0	1.39%	483.63	1.00	0.925	5	2.13%	0.60%	-0.78%
p03	641.19	648.60	1.16%	668.71	1.00	1.00	0	3.10%	654.34	1.00	0.95	2	2.05%	0.88%	-2.15%
p04	1001.04	1043.05	4.20%	1108.63	1.00	1.00	2	6.29%	1073.77	1.00	0.95	1	7.27%	2.95%	-3.14%
p05	750.03	775.13	3.35%	792.15	1.00	1.00	0	2.19%	778.85	1.00	0.975	0	3.84%	0.48%	-1.68%
p06	876.50	897.16	2.36%	966.85	1.00	1.00	0	7.77%	924.28	1.00	0.925	0	5.45%	3.02%	-4.40%
p07	881.97	899.96	2.04%	961.25	1.00	1.00	0	6.81%	940.16	1.00	0.975	16	6.60%	4.47%	-2.19%
p08	4371.66	4495.17	2.83%	5058.74	0.99	1.00	0	12.54%	4623.25	1.00	0.975	1	5.75%	2.85%	-8.61%
p09	3858.66	3962.64	2.69%	4273.86	1.00	1.00	0	7.85%	4056.49	1.00	0.975	15	5.13%	2.37%	-5.09%
p10	3629.60	3738.80	3.01%	3925.71	1.00	1.00	0	5.00%	3794.06	1.00	0.975	16	4.53%	1.48%	-3.35%
p11	3545.18	3612.26	1.89%	3815.84	1.00	1.00	0	5.64%	3677.32	1.00	0.975	1	3.73%	1.80%	-3.63%
p12	1318.95	1318.95	0.00%	1326.89	1.00	1.00	15	0.60%	1326.74	1.00	0.975	0	0.59%	0.59%	-0.01%
p13	1318.95	1318.95	0.00%	1326.71	1.00	0.95	0	0.59%	1326.68	1.00	1.00	0	0.59%	0.59%	0.00%
p14	1360.12	1360.12	0.00%	1361.93	1.00	0.90	0	0.13%	1361.50	1.00	0.975	0	0.10%	0.10%	-0.03%
p15	2505.42	2557.53	2.08%	2666.42	1.00	1.00	0	4.26%	2590.15	1.00	0.925	2	3.38%	1.28%	-2.86%
p16	2572.23	2587.86	0.61%	2616.13	1.00	0.975	0	1.09%	2616.13	1.00	0.975	0	1.71%	1.09%	0.00%
p17	2709.09	2714.66	0.21%	2718.27	1.00	1.00	24	0.13%	2718.27	1.00	1.00	24	0.34%	0.13%	0.00%
p18	3702.85	3812.30	2.96%	3841.51	1.00	1.00	0	0.77%	3833.52	1.00	0.975	20	3.53%	0.56%	-0.21%
p19	3827.06	3876.15	1.28%	3918.52	1.00	1.00	20	1.09%	3918.52	1.00	1.00	20	2.39%	1.09%	0.00%
p20	4058.07	4085.91	0.69%	4091.26	1.00	0.90	2	0.13%	4091.26	1.00	0.90	2	0.82%	0.13%	0.00%
p21	5474.84	5681.16	3.77%	5849.68	1.00	1.00	6	2.97%	5741.34	1.00	0.925	19	4.87%	1.06%	-1.85%
p22	5702.16	5808.74	1.87%	5864.13	1.00	0.975	29	0.95%	5864.13	1.00	0.975	29	2.84%	0.95%	0.00%
p23	6078.75	6140.01	1.01%	6147.81	1.00	0.90	0	0.13%	6147.47	1.00	0.90	0	1.13%	0.12%	-0.01%

TABLE B.6: Costs considering different instances with a medium demand variability

Inst.	BKS (1)	BDS-FC (2)	G. (2)-(1)	BDS-TEC (3)	BDS-R	BDS-K	BDS-T	G. (3)-(2)	BSS-TEC (4)	BSS-R	BSS-K	BSS-T	G. (4)-(1)	G. (4)-(2)	G. (4)-(3)
p01	576.87	580.49	0.63%	621.78	0.99	1.00	0	7.11%	607.92	1.00	0.925	0	5.38%	4.73%	-2.23%
p02	473.53	481.05	1.59%	495.72	0.99	1.00	0	3.05%	484.47	1.00	0.925	0	2.31%	0.71%	-2.27%
p03	641.19	648.80	1.19%	672.36	1.00	1.00	0	3.63%	663.64	1.00	0.925	9	3.50%	2.29%	-1.30%
p04	1001.04	1039.64	3.86%	1194.31	0.98	1.00	0	14.88%	1136.68	0.99	0.95	0	13.55%	9.33%	-4.83%
p05	750.03	775.40	3.38%	814.19	0.99	1.00	8	5.00%	797.13	1.00	0.95	1	6.28%	2.80%	-2.10%
p06	876.50	896.83	2.32%	991.64	0.98	1.00	1	10.57%	947.47	1.00	0.925	4	8.10%	5.65%	-4.45%
p07	881.97	901.93	2.26%	988.31	0.98	1.00	0	9.58%	975.88	0.99	0.975	0	10.65%	8.20%	-1.26%
p08	4371.66	4496.87	2.86%	5173.76	0.99	1.00	1	15.05%	4726.09	1.00	0.95	0	8.11%	5.10%	-8.65%
p09	3858.66	3981.77	3.19%	4290.58	0.99	1.00	0	7.76%	4116.24	1.00	0.95	28	6.68%	3.38%	-4.06%
p10	3629.60	3754.90	3.45%	4032.78	1.00	1.00	0	7.40%	3866.11	1.00	0.95	25	6.52%	2.96%	-4.13%
p11	3545.18	3607.04	1.75%	3953.10	0.99	1.00	0	9.59%	3767.31	1.00	0.925	10	6.27%	4.44%	-4.70%
p12	1318.95	1318.95	0.00%	1356.83	1.00	0.975	0	2.87%	1356.83	1.00	0.975	0	2.87%	2.87%	0.00%
p13	1318.95	1318.95	0.00%	1355.84	1.00	0.975	0	2.80%	1355.84	1.00	0.975	0	2.80%	2.80%	0.00%
p14	1360.12	1360.12	0.00%	1393.37	1.00	0.90	0	2.45%	1392.10	1.00	0.925	0	2.35%	2.35%	-0.09%
p15	2505.42	2549.17	1.75%	2722.47	0.99	1.00	0	6.80%	2635.44	1.00	0.90	0	5.19%	3.38%	-3.20%
p16	2572.23	2587.86	0.61%	2671.31	1.00	0.975	0	3.22%	2671.31	1.00	0.975	0	3.85%	3.22%	0.00%
p17	2709.09	2714.66	0.21%	2783.04	1.00	1.00	24	2.52%	2783.04	1.00	1.00	24	2.73%	2.52%	0.00%
p18	3702.85	3814.73	3.02%	3930.95	1.00	1.00	12	3.05%	3916.54	1.00	0.925	21	5.77%	2.67%	-0.37%
p19	3827.06	3875.40	1.26%	4002.83	1.00	1.00	28	3.29%	4002.83	1.00	1.00	28	4.59%	3.29%	0.00%
p20	4058.07	4085.91	0.69%	4187.05	1.00	0.90	2	2.48%	4187.05	1.00	0.90	2	3.18%	2.48%	0.00%
p21	5474.84	5690.22	3.93%	5891.98	1.00	1.00	19	3.55%	5870.96	1.00	0.925	17	7.24%	3.18%	-0.36%
p22	5702.16	5796.48	1.65%	5980.72	1.00	0.975	4	3.18%	5980.72	1.00	0.975	4	4.89%	3.18%	0.00%
p23	6078.75	6140.01	1.01%	6290.98	1.00	0.90	0	2.46%	6289.53	1.00	0.90	0	3.47%	2.44%	-0.02%

TABLE B.7: Costs considering different instances with a high demand variability

Inst.	BKS (1)	BDS-FC (2)	G. (2)-(1)	BDS-TEC (3)	BDS-R	BDS-K	BDS-T	G. (3)-(2)	BSS-TEC (4)	BSS-R	BSS-K	BSS-T	G. (4)-(1)	G. (4)-(2)	G. (4)-(3)
p01	576.87	587.18	1.79%	633.62	0.99	1.00	0	7.91%	620.93	1.00	0.925	0	7.64%	5.75%	-2.00%
p02	473.53	479.45	1.25%	491.96	0.99	1.00	12	2.61%	485.77	1.00	0.90	0	2.58%	1.32%	-1.26%
p03	641.19	649.87	1.35%	679.88	1.00	1.00	6	4.62%	671.69	1.00	0.925	0	4.76%	3.36%	-1.20%
p04	1001.04	1042.05	4.10%	1197.08	0.97	1.00	0	14.88%	1177.35	0.98	0.95	6	17.61%	12.98%	-1.65%
p05	750.03	777.41	3.65%	821.53	0.98	1.00	0	5.67%	806.97	0.99	0.95	0	7.59%	3.80%	-1.77%
p06	876.50	897.48	2.39%	1021.06	0.97	1.00	0	13.77%	971.84	0.99	0.925	10	10.88%	8.29%	-4.82%
p07	881.97	907.38	2.88%	1011.02	0.97	1.00	0	11.42%	991.80	0.97	1.00	2	12.45%	9.30%	-1.90%
p08	4371.66	4498.65	2.90%	5267.60	0.98	1.00	0	17.09%	4772.74	1.00	0.925	6	9.17%	6.09%	-9.39%
p09	3858.66	3962.30	2.69%	4398.51	0.99	1.00	22	11.01%	4185.01	1.00	0.95	17	8.46%	5.62%	-4.85%
p10	3629.60	3747.91	3.26%	4106.36	0.99	1.00	0	9.56%	3940.02	1.00	0.95	15	8.55%	5.13%	-4.05%
p11	3545.18	3625.26	2.26%	4010.29	0.99	1.00	0	10.62%	3788.72	1.00	0.925	11	6.87%	4.51%	-5.53%
p12	1318.95	1318.95	0.00%	1377.66	1.00	1.00	0	4.45%	1377.66	1.00	1.00	0	4.45%	4.45%	0.00%
p13	1318.95	1318.95	0.00%	1376.28	0.99	1.00	0	4.35%	1376.28	0.99	1.00	0	4.35%	4.35%	0.00%
p14	1360.12	1360.12	0.00%	1417.01	0.99	0.90	0	4.18%	1414.06	0.99	0.925	0	3.97%	3.97%	-0.21%
p15	2505.42	2553.90	1.93%	2755.74	0.98	1.00	0	7.90%	2673.11	1.00	0.90	2	6.69%	4.67%	-3.00%
p16	2572.23	2590.77	0.72%	2712.30	0.99	0.975	30	4.69%	2712.30	0.99	0.975	30	5.45%	4.69%	0.00%
p17	2709.09	2714.66	0.21%	2823.97	1.00	0.90	29	4.03%	2823.97	1.00	0.90	29	4.24%	4.03%	0.00%
p18	3702.85	3813.22	2.98%	4005.61	0.99	0.975	0	5.05%	3971.55	1.00	0.925	24	7.26%	4.15%	-0.85%
p19	3827.06	3876.15	1.28%	4054.76	0.99	1.00	0	4.61%	4054.76	0.99	1.00	0	5.95%	4.61%	0.00%
p20	4058.07	4085.91	0.69%	4252.71	0.99	0.90	2	4.08%	4252.71	0.99	0.90	2	4.80%	4.08%	0.00%
p21	5474.84	5677.62	3.70%	5974.19	0.99	1.00	20	5.22%	5957.66	0.99	0.925	13	8.82%	4.93%	-0.28%
p22	5702.16	5812.03	1.93%	6079.27	0.99	0.975	5	4.60%	6079.27	0.99	0.975	5	6.61%	4.60%	0.00%
p23	6078.75	6140.01	1.01%	6388.07	1.00	0.90	0	4.04%	6386.58	1.00	0.975	0	5.06%	4.02%	-0.02%

TABLE B.8: Tabulated summary of the results

Scenario	G. BDS-FC - BKS	G. BDS-TEC - BDS-FC	G. BSS-TEC - BKS	G. BSS-TEC - BDS-FC	G. BSS-TEC - BDS-TEC
Var: $0.10E[D_i]$	1.83%	3.10%	3.12%	1.26%	−1.69%
Var: $0.50E[D_i]$	1.83%	5.89%	5.53%	3.62%	−2.06%
Var: $1.00E[D_i]$	1.74%	7.48%	7.12%	5.27%	−1.97%

variability of the scenario. Next two columns quantify the gaps between the BSS-TEC and its lower bounds, the BKS and the BDS-FC. They both increase as the variability of the scenario gets higher. Finally, the last gap shows the benefit of using our simheuristic approach. Thus, it can be concluded that the higher the variability the higher the benefit.

Here we present a risk analysis in which the four best stochastic solutions and the best deterministic solution are compared. It is illustrated on a specific case, the instance p09 with high variance. Thus, Figure B.5 shows a boxplot of the total costs obtained by means of MCS. The means of the observations and their corresponding confidence intervals are displayed in Figure B.6. In Figure B.7, the empirical Cumulative Distributions Functions (CDFs) for the best deterministic and stochastic solutions are drawn. It can be stated that the variability of total costs associated to the best deterministic solution is the highest (Figure B.5), and all distributions present a positive skew. Figure B.6 shows that the confidence intervals of the means related to the best stochastic solutions do not overlap, which indicates that the differences are statistically significant. Note that the big sample size of each group (2,000) leads to relatively narrow intervals. In Figure B.7, the probability distribution function of the best stochastic solution is above the other almost for the entire domain. In other words, the probability of having a total cost equal to or lower than a given value is usually higher with this solution. As a consequence, a risk-averse decision-maker would prefer it. Nevertheless, the minimum values are provided by the deterministic solution, which makes sense since this solution will be the one selected in scenarios where the customer demands are similar to the corresponding mean of the distributions.

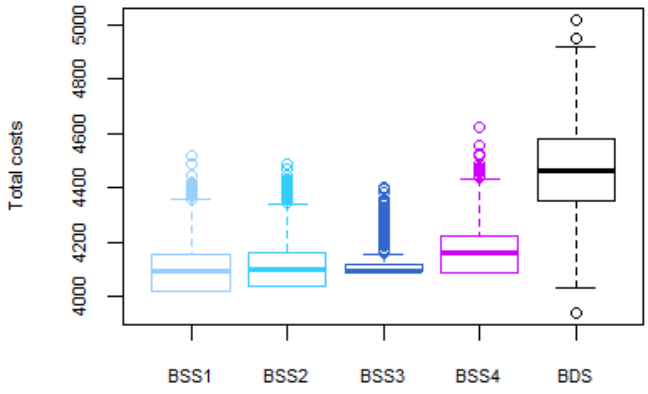


FIGURE B.5: Boxplots of best solutions for instance p09 with high variability

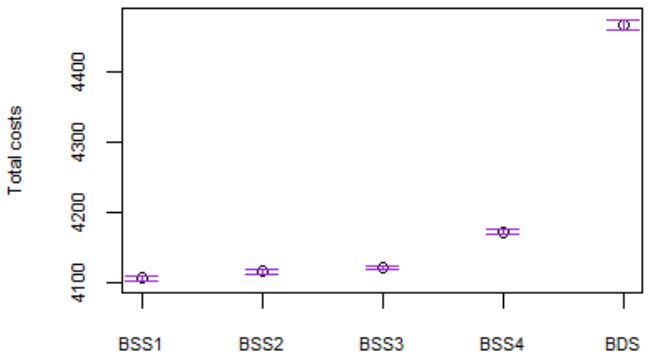


FIGURE B.6: Expected total cost and CIs of best solutions for instance p09 with high variability

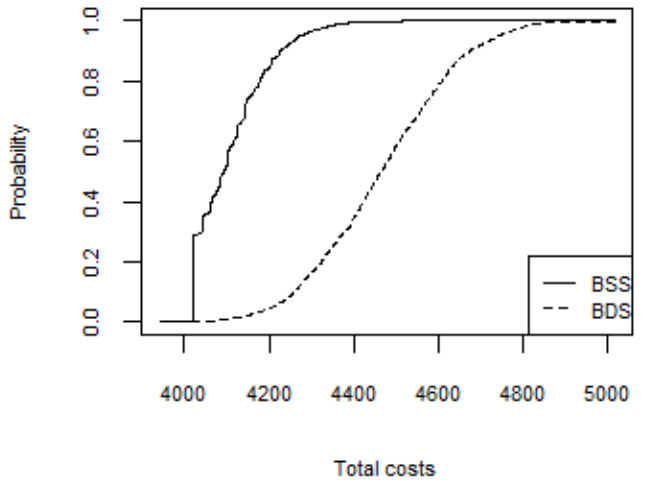


FIGURE B.7: CDFs of best deterministic and stochastic solutions for instance p09 with high variability

Regarding the comparison among algorithms, an analysis of variance (Montgomery, 2008) is performed by using the gaps between the BSS-TEC and the BKS for the second scenario (i.e., medium level of variability). Since the normality assumption is not fully satisfied, we apply the Kruskal-Wallis test by ranks. The p-value is 0.90, which indicates that the differences between medians are not statistically significant. Boxplots in Figure B.8 show the distributions of the gaps for each algorithm, revealing that the simheuristic based on the LNS metaheuristic seems to provide slightly better results on average.

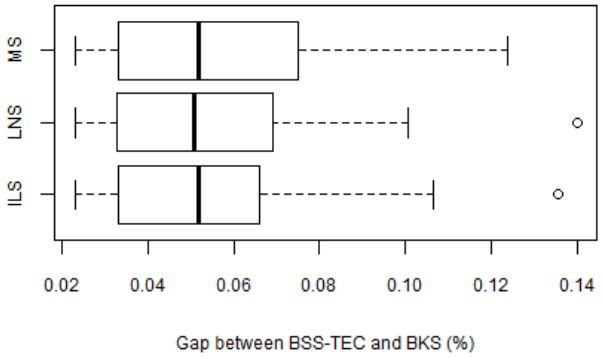


FIGURE B.8: Comparison among simheuristic algorithms

7. Conclusions and future research

In this paper we have presented a simheuristic algorithm for solving the Stochastic and Capacitated Multi-Depot Vehicle Routing Problem (MDVRP). It is based on the idea that this problem can be analyzed as a generalization of the deterministic version, which allows to combine efficient metaheuristic algorithms with simulation to solve the stochastic version. We propose a two-stage methodology. Initially, routes are designed considering the expected demands and a vehicle capacity lower than the real one. This capacity surplus is to be employed when demands are higher than expected. Then, in the routing phase, the actual routes are specified. In case of a route failure, the vehicle returns to the depot to reload and continues the planned route. In order to reduce the risk of a route failure, preventive re-stocking is carried out when the remaining vehicle capacity is lower than the expected demand of the next customer.

Our methodology combines a metaheuristic framework, which guides the exploration of the search space, with Monte Carlo simulation techniques, which assess the quality of the promising solutions. Using simulation techniques, it is possible to perform a risk analysis in order to take into account the risk-aversion of the decision-maker for selecting the best solution. The efficiency of our algorithm is evaluated throughout a set of experiments considering different levels of demand variability, comparing three algorithms relying on different metaheuristics and providing both lower and upper bounds. There are very few works addressing the Stochastic MDVRP and they consider uncapacitated depots. In addition, these works usually make strong assumptions about the probability distribution of the demands and their independence. Relying on the simheuristic framework, we relax these assumptions, just requiring historical data or theoretical distributions with an existing mean. Thus, our methodology aims to deal with more realistic scenarios. Our results show the importance of tackling Combinatorial Optimization Problems taking into account the uncertainty that characterizes our world, avoiding the traditional approach that assumes that these problems are (or can be solved as if they were) deterministic.

In this work it has been assumed that all customers have to be visited and their demands are required to be completely satisfied. Thus, we have focused on constructing ‘robust’ or ‘reliable’ solutions (i.e., solutions with lower expected cost but also with lower variability or risk). In our view, these robust solutions can reduce the intensity/impact of corrective actions over the scheduled distribution plan and, therefore, they constitute a first step towards a complete – and yet to be fully explored – framework that also takes into account these corrective actions as a response to stochastic and dynamic environments. Similarly, there are other realistic and richer extensions that could be studied such as the Stochastic and Capacitated MDVRP with Heterogeneous Depots, Multiple Products, or Stochastic Distances.

Acknowledgements

References

Bae, S.-T., H.S. Hwang, G.-S. Cho, and M.-J. Goan (2007). "Integrated GA-VRP solver for multi-depot system". In: *Computers & Industrial Engineering* 53, pp. 233–240.

Bastian, C. and A. H. G. R. Kan (1992). "The stochastic vehicle routing problem revisited". In: *European Journal of Operational Research* 56, pp. 407–412.

Bertsimas, D. J. (1988). "Probabilistic combinatorial optimization problems". PhD thesis. Operations Research Center, Massachusetts Institute of Technology.

Bianchi, L., M. Birattari, M. Chiarandini, M. Manfrin, M. Mastrolilli, L. Paquete, O. Rossi-Doria, and T. Schiavinotto (2006). "Hybrid metaheuristics for the vehicle routing problem with stochastic demands". In: *Journal of Mathematical Modelling and Algorithms* 5, pp. 91–110.

Bianchi, L., D. Marco, L. M. Gambardella, and W. J. Gutjahr (2009). "A survey on metaheuristics for stochastic combinatorial optimization". In: *Natural Computing* 8, pp. 239–287.

Cabrera, G., A. A. Juan, D. Lázaro, J. M. Marquès, and I. Proskurnia (2014). "A simulation-optimization approach to deploy Internet services in large-scale systems with user-provided resources". In: *Simulation* 90.6, pp. 644–659.

Caceres, J., P. Arias, D. Guimarans, D. Riera, and A. A. Juan (2014). "Rich vehicle routing problem: a survey". In: *ACM Computing Surveys* 47.2. ISSN: 0360-0300.

Chan, Y., W.B. Carter, and M.D. Burnes (2001). "A hybrid algorithm for multi-depot vehicle routing problem". In: *Computers & Operations Research* 28.8, pp. 803–826.

Chao, I., B. Golden, and E. Wasil (1993). "A new heuristic for the multi-depot vehicle routing problem that improves upon best known solutions". In: *American Journal of Mathematical and Management Sciences* 13.3-4, pp. 371–406.

Chen, P. and X. Xu (2008). "A hybrid algorithm for multi-depot vehicle routing problem". In: *IEEE International Conference on Service Operations and Logistics, and Informatics*, pp. 2031–2034.

Christofides, N. and S. Eilon (1969). "An algorithm for the vehicle-dispatching problem". In: *Operational Research Quaterly* 20, pp. 309–318.

Clarke, G. and J. Wright (1964). "Scheduling of vehicles from a central depot to a number of delivering points". In: *Operations Research* 12, pp. 568–581.

Cordeau, J.F. and M. Maischberger (2012). "A parallel iterated tabu search heuristic for vehicle routing problems". In: *Computers & Operations Research* 39.9, pp. 2033–2050.

Cordeau, J.F., M. Gendreau, and G. Laporte (1997). "A tabu search heuristic for periodic and multidepot vehicle routing problems". In: *Networks* 30.2, pp. 105–119.

Crevier, B., J. F. Cordeau, and G. Laporte (2007). "The multi-depot vehicle routing problem with inter-depot routes". In: *European Journal of Operational Research* 176, pp. 756–773.

Dondo, R. and J. Cerdá (2009). "A hybrid local improvement algorithm for large-scale multi-depot vehicle routing problems with time windows". In: *Computers & Chemical Engineering* 33, pp. 513–530.

Dror, M. and P. Trudeau (1986). "Stochastic vehicle routing with modified savings algorithms". In: *European Journal of Operational Research* 23, pp. 228–235.

Erbao, C. and L. Mingyong (2009). "A hybrid differential evolution algorithm to vehicle routing problem with fuzzy demands". In: *Journal of Computational and Applied Mathematics* 231, pp. 302–310.

Escobar, J. W., R. Linfati, P. Toth, and M. G. Baldoquin (2014). "A hybrid granular tabu search algorithm for the multi-depot vehicle routing problem". In: *Journal of Heuristics* 20, pp. 483–509.

Gendreau, M., G. Laporte, and R. Séguin (1995). "An exact algorithm for the vehicle routing problem with stochastic demands and customers". In: *Transportation Science* 29, pp. 143–155.

— (1996). "Stochastic vehicle routing with modified savings algorithms". In: *European Journal of Operational Research* 88, pp. 3–12.

Gillett, B.E. and J.G. Johnson (1976). "Multi-terminal vehicle-dispatch algorithm". In: *Omega* 4.6, pp. 711–718.

Golden, B.L., T.L. Magnanti, and H.Q. Nguyen (1977). "Implementing vehicle routing algorithms". In: *Networks* 7.2, pp. 113–148.

Gonzalez, S., D. Riera, A. A. Juan, M. G. Elizondo, and P. Fonseca (2012). "Proceedings of the 2012 Winter Simulation Conference". In: *SIM-RANDSHARP: a hybrid algorithm for solving the arc routing problem with stochastic demands*. Ed. by C. Laroque, J. Himmelspach, R. Pasupathy, O. Rose, and A. M. Uhrmacher, pp. 1–12.

Gonzalez, S., A. A. Juan, D. Riera, M. Elizondo, and J. Ramos (2016). "A simheuristic algorithm for solving the arc routing problem with stochastic demands". In: *Journal of Simulation*.

Goodson, J. C. (2015). "A priori policy evaluation and cyclic-order-based simulated annealing for the multi-compartment vehicle routing problem with stochastic demands". In: *European Journal of Operational Research* 241, pp. 361–369.

Gulczynski, D., B. L. Golden, and E. Wasil (2011). "The multi-depot split delivery vehicle routing problem: An integer programming-based heuristic, new test problems, and computational results". In: *Computers & Industrial Engineering* 61, pp. 794–804.

Ho, W., G.T.S. Ho, P. Ji, and H.C.W. Lau (2008). "A hybrid genetic algorithm for the multi-depot vehicle routing problem". In: *Engineering Applications of Artificial Intelligence* 21.4, pp. 548–557.

Ismail, Z. and I. Irhamah (2008). "Solving the vehicle routing problem with stochastic demands via hybrid genetic algorithm-tabu search". In: *Journal of Mathematics and Statistics* 4, pp. 161–167.

Jaillet, P. and M R Wagner (2008). "Online Vehicle Routing Problems: A Survey". In: *The Vehicle Routing Problem: Latest Advances and New Challenges*. Ed. by B. L. Golden, S. Raghavan, and E. A. Wasil. New York: Springer, pp. 221–238.

Juan, A. A., J. Faulin, R. Ruiz, B. Barrios, and S. Caballe (2010). "The SR-GCWS hybrid algorithm for solving the capacitated vehicle routing problem". In: *Applied Soft Computing* 10.1, pp. 215–224.

Juan, A. A., J. Faulin, J. Jorba, D. Riera, D. Masip, and B. Barrios (2011a). "On the use of Monte Carlo simulation, cache and splitting techniques to improve the Clarke and Wright savings heuristics". In: *Journal of the Operational Research Society* 62, pp. 1085–1097.

Juan, A. A., J. Faulin, S. Grasman, D. Riera, J. Marull, and C. Mendez (2011b). "Using safety stocks and simulation to solve the vehicle routing problem with stochastic demands". In: *Transportation Research Part C: Emerging Technologies* 19.5, pp. 751–765.

Juan, A. A., J. Faulin, J. Jorba, J. Caceres, and J. M. Marques (2013b). "Using parallel & distributed computing for real-time solving of vehicle routing problems with stochastic demands". In: *Annals of Operations Research* 207.1, pp. 43–65.

Juan, A. A., B. Barrios, E. Vallada, D. Riera, and J. Jorba (2014a). "A simheuristic algorithm for solving the permutation flow shop problem with stochastic processing times". In: *Simulation Modelling Practice and Theory* 46, pp. 101–117.

Juan, A. A., S. E. Grasman, J. Caceres-Cruz, and T. Bektaş (2014b). "A simheuristic algorithm for the single-period stochastic inventory-routing problem with stock-outs". In: *Simulation Modelling Practice and Theory* 46, pp. 40–52.

Juan, A. A., J. Faulin, S. Grasman, M. Rabe, and G. Figueira (2015a). "A review of simheuristics: extending metaheuristics to deal with stochastic optimization problems". In: *Operations Research Perspectives* 2, pp. 62–72.

Juan, A. A., I. Pascual, D. Guimarans, and B. Barrios (2015c). "Combining biased randomization with iterated local search for solving the multidepot vehicle routing problem". In: *International Transactions in Operational Research* 22.4, pp. 647–667.

Karakatic, S. and V. Podgorelec (2015). "A survey of genetic algorithms for solving multi depot vehicle routing problem". In: *Applied Soft Computing* 27, pp. 519–532.

Kenyon, A. S. and D. P. Morton (2003). "Stochastic vehicle routing with random travel times". In: *Transportation Science* 37, pp. 69–82.

Kouvelis, P. and G. Yu (1997). *Robust discrete optimization and its applications*. Dordrecht, The Netherlands: Kluwer Academic Publishers.

Laporte, G., F. Louveaux, and H. Mercure (1992). "The vehicle routing problem with stochastic travel times". In: *Transportation Science* 26, pp. 161–170.

— (1994). "An exact solution for the a priori optimization of the probabilistic traveling salesman problem". In: *Operations Research* 42, pp. 543–549.

Li, J., P.M. Pardalos, H. Sun, J. Pei, and Y. Zhang (2015). "Iterated local search embedded adaptive neighborhood selection approach for the multi-depot vehicle routing problem with simultaneous deliveries and pickups". In: *Expert Systems with Applications* 42.7, pp. 3551–3561.

Lourenço, H.R., O.C. Martin, and T. Stützle (2010). "Iterated local search: framework and applications". In: *Handbook of Metaheuristics*. Ed. by M. Gendreau and J.-Y. Potvin. Vol. 146. International Series in Operations Research & Management Science. Springer US, pp. 363–397.

Martí, R., M. G. C. Resende, and C. Ribeiro (2013). "Multi-start methods for combinatorial optimization". In: *European Journal of Operational Research* 226.1, pp. 1–8.

Mirabi, M., S.M.T. Fatemi-Ghomi, and F. Jolai (2010). "Efficient stochastic hybrid heuristics for the multi-depot vehicle routing problem". In: *Robotics and Computer-Integrated Manufacturing* 26.6, pp. 564–569.

Moghaddam, B. F., R. Ruiz, and S. J. Sadjadi (2012). "Vehicle routing problem with uncertain demands: An advanced particle swarm algorithm". In: *Computers & Industrial Engineering* 62, pp. 306–317.

Montgomery, D. C. (2008). *Design and analysis of experiments*. 8th ed. John Wiley & Sons.

Montoya-Torres, J., J. Lopez, S. Nieto, H. Felizzola, and N. Herazo-Padilla (2015). "A literature review on the vehicle routing problem with multiple depots". In: *Computers & Industrial Engineering* 79, pp. 115–129.

Nikolaev, A. G. and S. H. Jacobson (2010). "Simulated annealing". In: *Handbook of metaheuristics*. Springer, pp. 1–39.

Pisinger, D. and S. Ropke (2007). "A general heuristic for vehicle routing problems". In: *Computers and Operations Research* 34.8, pp. 2403–2435.

— (2010). "Large neighborhood search". In: *Handbook of metaheuristics*. Springer, pp. 399–419.

Raft, O.M. (1982). "A modular algorithm for an extended vehicle scheduling problem". In: *European Journal of Operational Research* 11, pp. 67–76.

Renaud, J., G. Laporte, and F.F. Boctor (1996). "A tabu search heuristic for the multi-depot vehicle routing problem". In: *Computers & Operations Research* 23.3, pp. 229–235.

Salhi, S. and M. Sari (1997). "A multi-level composite heuristic for the multi-depot vehicle fleet mix problem". In: *European Journal of Operational Research* 103, pp. 95–112.

Stewart, Jr. W. R. and B.L. Golden (1983). "Stochastic vehicle routing: A comprehensive approach". In: *European Journal of Operational Research* 14, pp. 371–385.

Surekha, P. and S. Sumathi (2011). "Solution to MDVRP using genetic algorithms". In: *World Applied Programming* 1, pp. 118–131.

Talbi, E.-G. (2009). *Metaheuristics: From design to implementation*. New Jersey: John Wiley & Sons.

Tan, K. C., C. Y. Cheong, and C. K. Goh (2007). "Solving multiobjective vehicle routing problem with stochastic demand via evolutionary computation". In: *Discrete optimization* 177, pp. 813–839.

Tatarakis, A. and I. Minis (2009). "Stochastic single vehicle routing with a predefined customer sequence and multiple depot returns". In: *European Journal of Operational Research* 197, pp. 557–571.

Tauhid, S., Z. Jannat, and F. Mahmud (2012). "A novel three-phase approach for solving multi-depot vehicle routing problem with stochastic demand". In: *Algorithms Research* 1, pp. 15–19.

Thangiah, S. R. and S. Salhi (2001). "Genetic clustering: an adaptive heuristic for the multi-depot vehicle routing problem". In: *Applied Artificial Intelligence* 15.4, pp. 361–383.

Tillman, F. A. (1969). "The multiple terminal delivery problem with probabilistic demands". In: *Transportation Science* 3, pp. 192–204.

Tillman, F. A. and T. M. Cain (1972). "An upper bound algorithm for the single and multiple terminal delivery problem". In: *Management Science* 18.11, pp. 664–682.

Vidal, T., T G Crainic, M. Gendreau, N. Lahrichi, and W. Rei (2012). "A hybrid genetic algorithm for multi-depot and periodic vehicle routing problems". In: *Operations Research* 60, pp. 611–624.

Wu, T.-H., C. Low, and J.-W. Bai (2002). "Heuristic solutions to multi-depot location-routing problems". In: *Computers & Operations Research* 29.10, pp. 1393–1415.

Yang, W. H., K. Mathur, and R. H. Ballou (2000). "Stochastic vehicle routing problem with restocking". In: *Transportation Science* 34, pp. 99–112.

B.4 Metaheuristics for rich portfolio optimisation and risk management: Current state and future trends

Jana Doering[1], Laura Calvet[2], Renatas Kizys[3], Angel A. Juan[2], Àngels Fitó[1]

1. Economics and Business Department, Open University of Catalonia – IN3,
Av. Tibidabo 39-43, 08035 Barcelona, Spain
Tel.: +34-603-894448
e-mail: {jdoering, afitob}@uoc.edu
2. Computer Science Dept., Universitat Oberta de Catalunya – IN3,
Av. Carl Friedrich Gauss 5, 08060 Castelldefels, Spain
e-mail: {lcalvetl, ajuanp}@uoc.edu
3. Subject Group of Economics and Finance, Portsmouth Business School, University of Portsmouth,
Richmond Building, Portland Street, Portsmouth PO1 3DE, Hampshire, UK
e-mail: renatas.kizys@port.ac.uk

Abstract

Computational finance is an emerging application field of metaheuristic algorithms. In particular, these optimisation methods are becoming the solving approach alternative when dealing with realistic versions of several decision-making problems in finance, such as rich portfolio optimisation and risk management. This paper reviews the scientific literature on the use of metaheuristics for solving NP-hard versions of these combinatorial optimisation problems and illustrates their capacity to provide high-quality solutions under scenarios considering realistic constraints. The paper contributes to the existing literature in three ways. Firstly, it reviews the literature on metaheuristic optimisation applications for portfolio and risk management in a systematic way. Secondly, it identifies the linkages between portfolio optimisation and risk management and presents a unified view and classification of both problems. Finally, it outlines the trends that have gradually become apparent in the literature and will dominate future research in order to further improve the state-of-the-art in this knowledge area.

Keywords: portfolio optimisation, risk management, combinatorial optimisation, metaheuristics.

1. Introduction

Since the last century, the direct relationship between financial decisions and wealth creation through capital accumulation and economic development has been widely accepted (Patrick, 1966). Thus, investments play an essential role in improvements of welfare standards. This striving for improvement is represented through the formulation of optimisation problems for most of the questions in financial economics. Traditionally, exact methods have been employed in determining optimal solutions to these. These methods, however, present some limitations when solving realistic and large-scale combinatorial optimisation problems (COPs) of NP-hard nature, since under these circumstances they require either the use of simplifying (non-realistic) assumptions or extraordinarily long computing times. Because this approach neglects depicting the complex intricacies of the real-life problems that decision-makers face in their everyday actions, the results are predominantly not transferrable to real-life operations without reservations. Furthermore, the current internationalisation and integration of financial markets and institutions has caused financial decision-making processes to become even more complex, both in terms of associated constraints as well as in terms of the instances to solve. Advances in Operations Research and Computer Science have brought forward new solution approaches in optimisation theory, such as heuristics and metaheuristics (Boussaïd et al., 2013). While the former are experience-based procedures, which usually provide 'good' solutions in short computing times, metaheuristics are general templates that can easily be tailored to address a wide range of problems. They have shown to provide near-optimal solutions in reasonable computing times to problems for which traditional methods are not applicable (Michalewicz and Fogel, 2013). Since they usually require relatively little computational time, metaheuristics constitute an attractive alternative for problem solving in several knowledge areas in which real-time decisions are required. Among others, Talbi (2009)

provide an excellent overview of metaheuristic methodologies and their applications. In particular, applications of metaheuristics in the financial sector are presented in Gilli et al. (2011). While metaheuristics do not guarantee finding a globally optimal solution, Gilli and Schumann (2012) point out that the goal of optimisation in most real-life instances is not to provide an optimal solution, but one that fulfils the decision-maker's objectives to a highly satisfactory extent. Hence, these authors promote the use of metaheuristic approaches in practical applications. In effect, with respect to exact methods that provide an optimal solution to a simplified model of a real-life problem, metaheuristics can provide a near-optimal solution to a realistic model of the same problem, which might be preferable for most decision-makers. The contributions of this work to the existing literature are threefold. Firstly, it reviews the literature on metaheuristic optimisation applications for portfolio and risk management in a systematic way. This classification, in conjunction with the corresponding subproblems, is depicted in Fig. B.1. For investment decisions it is indicated whether the corresponding problem refers to an active or passive strategy. Further, an exemplary recent paper is provided for each subproblem.

As second contribution, the work identifies the linkages between portfolio optimisation and risk management and presents a unified view and classification of these problems. It is expected that the revocation of the strict classification of financial COPs can lead to a methodological transfer of knowledge in between different applications that enable more effective and efficient selections of decision-makers. Finally, the work also outlines the trends that have gradually become apparent in the literature and are expected to dominate future research in this knowledge area. Table B.1 presents an overview of these trends and challenges classified into problem-specific and methodology-specific dimensions.

The remainder of the paper is structured as follows: Section 2 presents the research methodology and an overview of recent publications. Section 3 consists of a short overview of metaheuristics for those readers who are less familiar with these methods. Following this, a review of the recent literature on portfolio optimisation and the corresponding subproblems is presented in Section 4, while Section 5 reviews the research on risk management problems. Further, the linkage between the two is discussed in Section 6. Future trends in the application of metaheuristics in the areas of portfolio optimisation and risk management are analysed in Section 7. Finally, Section 8 highlights the main findings and contributions of this work and concludes it.

2. Review Strategy

The increasing popularity of the application of metaheuristics to portfolio optimisation problems (POPs) and risk management problems (RMPs) is depicted below in Fig. B.2 based on Scopus-indexed publications that explicitly consider metaheuristics as an approach in solving different financial COPs. In the case of 2016, only the first semester (2016-1) has been considered.

The search for POPs was conducted by examining the articles that explicitly consider portfolio optimisation (or the American English equivalent), index tracking or project selection in the abstract, title or keywords and make use of metaheuristics. For risk management problems, the search terms were bankruptcy, credit risk or stock or foreign exchange trading. In the case of portfolio optimisation, it becomes obvious that the trend in publications is increasing, i.e., metaheuristics have received increased attention as solving approaches. This has previously been predicted by researchers due to their power in obtaining high quality solutions to many real world complex problems (Osman and Kelly, 1996). More specifically, continuing increases in computing power, the advancement of metaheuristic frameworks and parallelisation strategies favour these methodologies when dealing with NP-hard financial COPs. On the contrary, risk management problems seem to have received much less attention. These proportions are broken down in Fig. B.3, which shows that traditional portfolio optimisation represents the majority of metaheuristic applications.

One of the major contributions of this work is to discuss the idea that most risk management variants are strongly correlated with portfolio optimisation, i.e., that risk management problems can oftentimes be partially expressed as portfolio optimisation problems. To exemplify this assumption imagine a decision-maker in a loan decision process who is choosing a portfolio of successful applicants from a pool of potential loan receivers based on his acceptance criteria and budget. Accordingly, it is possible to transfer methodological knowledge from the well-studied portfolio optimisation problem to the less explored area of risk management problems.

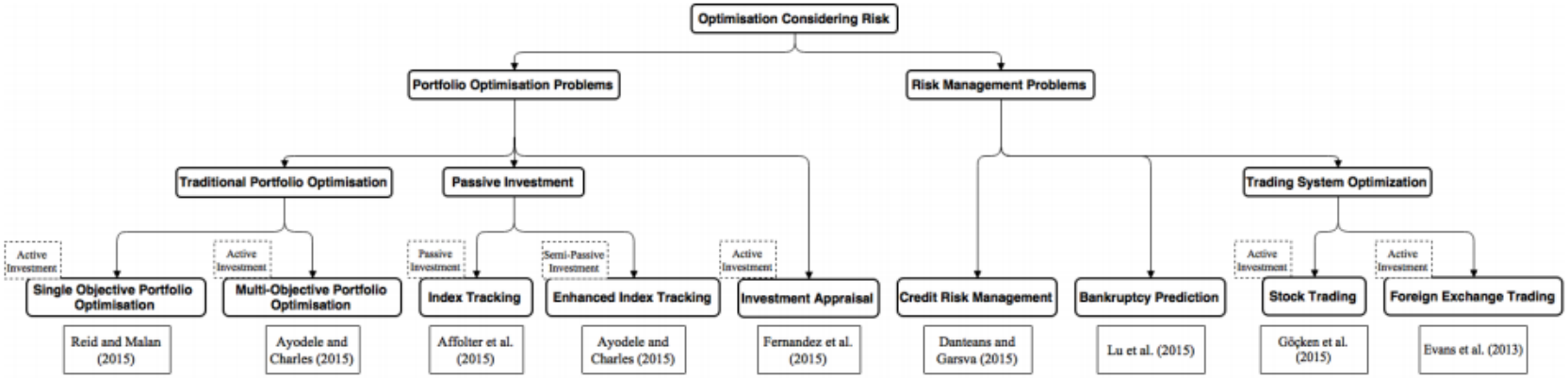


FIGURE B.1: Classification of two core areas regarding the use of metaheuristics in finance

TABLE B.1: Open research challenges associated with portfolio optimisation and risk management problems

Dimension		Research trends and challenges
Problem-specific	Realistic problem modelling	(1) Deviating from the traditional formulation, more accurate risk measures – such as value-at-risk variations – are to be evaluated with regard to their ability in improving the depiction of portfolio risk. (2) Hybridisations of simulation and optimisation should be employed to include in the optimisation model the macro- and micro-level uncertainty of financial markets.
	Problem complexity	(1) The introduction of additional required constraints and a more narrow execution of traditional constraints in a uniform way are yet to be presented. (2) The internationalisation and integration of financial markets call for the inclusion of an extended asset pool in portfolio optimisation problems.
Methodology-specific	Computational times	(1) The increasing complexity of the problem modelling calls for faster metaheuristic approaches. (2) Especially for large-scale problems, distributed and parallel computing techniques could be explored for real-time problem solving.
	Methodological complexity	(1) The predominance of population-based metaheuristics is not uniformly justified by the quality of the results; thus single-point metaheuristic approaches can be further explored. (2) The hybridisation of methodologies is a clear trend; however, this hybridisation should be done with care to avoid developing methods of increasing complexity that are difficult to reproduce in practice.

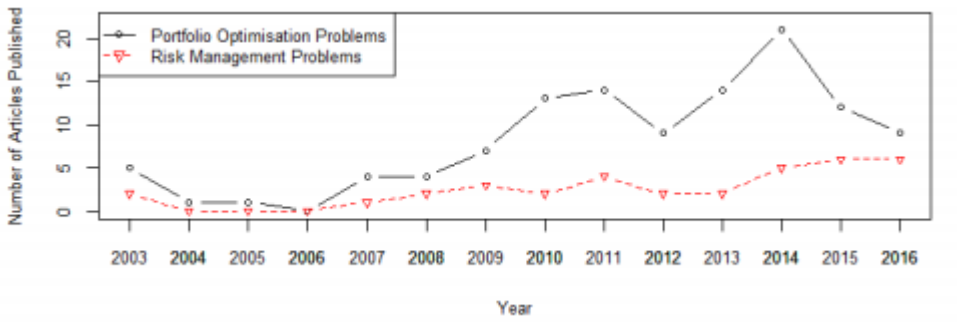


FIGURE B.2: Scopus-indexed publications applying metaheuristics to POPs and RMPs for the period 2003 to 2016-1

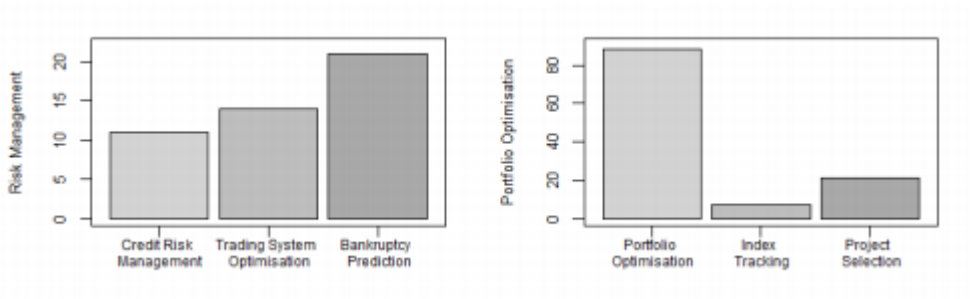


FIGURE B.3: Number of publications on the respective subproblems

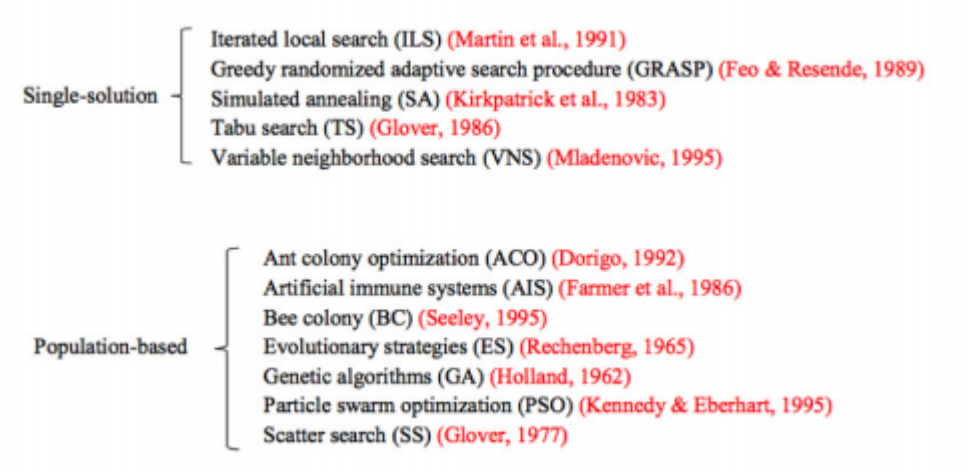


FIGURE B.4: Scheme of the most popular traditional metaheuristics

3. An Overview of Metaheuristics

Heuristics may be described as intelligent search strategies for solving problems (Pearl, 1984). They tend to be used in applications where exact methods fail to find a solution to a computationally hard problem and to speed up the search for high-quality solutions. Despite not guaranteeing optimality, heuristics have been extensively employed due to their high number of successful applications. Their main disadvantage is that most are problem- or even instance-dependent (Asta, 2015). As a consequence, considerable efforts to adapt them for addressing different problems or instances are needed.

Metaheuristics are intended to overcome this drawback. The term, first introduced by Glover (1986), can be described as a set of guidelines or strategies to develop heuristic optimisation algorithms. It is important to note that while metaheuristics (as frameworks) are domain-independent, their implementation is domain-specific. According to Feo and Resende (1995), the effectiveness of these methods greatly depends on their ability to adapt to a specific instance to solve, to avoid getting stuck in local optima, and to exploit the structure of a problem. The authors discuss the relevant role of restart procedures, controlled randomisation, efficient data structures, and pre-processing. The popularity of metaheuristics has grown rapidly among both the scientific community and practitioners. Research fields in which they are commonly and highly successfully employed include logistics and transportation, finance, machine learning, computer vision, cryptology, and healthcare sciences.

Metaheuristics can be classified into population-based metaheuristics, which work with a set of individual solutions that form a population, and single-solution metaheuristics, which maintain a single solution. While the former focus on exploration (diversification), searching a relatively large area of the search space, the latter centre on exploitation (intensification), applying local search within a limited region. Fig. B.4 lists the most employed metaheuristics with regards to this classification and includes the references of the first applications.

Despite this list being relatively short, a vast variety of metaheuristic methodologies exist, especially through hybridisations, and frequently, research communities focus only on a subset. The

TABLE B.2: The application of traditional metaheuristics and hybridisation to sub-problems of portfolio optimisation

Optimisation Problem	Single-solution search				Population-based search										Hybrid
	SA	TS	FD	SD	GA	FA	ACO	DE	EA	ABC	PSO	IWO	AIS	SS	
Single-objective portfolio optimisation	2	3	1	1	2				1		3				3
Multi-objective portfolio optimisation					2	2		2	1	1	4				2
Index tracking	1				2			2	3			1			3
Enhanced index tracking	1	1	1	1	2			1			1		2		1
Project selection		3			2		6				1			2	5

successful application of metaheuristics has led to an increase in interest in improvements and new developments of these methodologies in the academic community. However, more recently, while recognising the high value of many modern contributions, researchers occasionally criticise the lack of a scientific base, which is replaced by the most diverse metaphors (Sörensen, 2015), and leads to irreproducibility of the results and thus lack of reliability of the computational experiments. As we will show, this has also been the case for individual papers reviewed in this article. This is why the application of simplified, reproducible metaheuristics is a pressing open line of further research. Finally, we refer the reader interested in an extensive review of metaheuristics to Talbi (2009) and Gendreau and Potvin (2010).

4. Portfolio Optimisation

Since Markowitz (1952) developed the portfolio optimisation theory centred around the mean-variance approach, the academic community has been highly engaged in advancing the tools for portfolio optimisation. The theory is based on two constituting assumptions, namely: *(i)* the financial investors being concerned with the expected returns; and *(ii)* the risk of their respective investment. It is thus the goal to minimise the level of risk expressed through the portfolio variance for a given expected return level, resulting in the so-called unconstrained efficient frontier, from which the portfolio choice is determined by the risk awareness of the investor. This established the portfolio optimisation problem, which is a strategy of: *(i)* selection of financial assets; and *(ii)* determination of the optimal weights allocated to those assets that results in a desired portfolio return and associated minimum level of risk. Based on the investor's involvement with the asset selection, two types of investment management strategies can be identified. On the one hand, active investment strategies aim at beating market returns. On the other hand, passive investment strategies aim at replicating a benchmark index. This strategy has become specifically popular with equity funds and although it is originally based on the efficient market hypothesis, passively indexed funds can still outperform active funds and have shown to do so on average due to the increased management costs of active funds in the presence of market failures (Malkiel, 2003). According to these conclusions, index replication is not solely a hedging strategy, but provides stable profitability.

Table B.2 presents a summary of the metaheuristics applied to each of the problems reviewed in this section: single-objective portfolio optimisation, multi-objective portfolio optimisation, index tracking, enhanced index tracking, and project portfolio selection. The number of articles found on each topic and metaheuristic is included inside each cell. The classical portfolio optimisation is an active investment strategy, particularly when active re-balancing of the portfolio takes place in multi-period observations and, by its nature, investment appraisal requires the active selection of project portfolios. Index tracking is traditionally a passive strategy, while enhanced index tracking involves active management to some extent. Different metaheuristics can be classified with respect to different characteristics. As previously pointed out, they are classified in the following depending on whether they conduct a population-based or a single-solution search. While the latter can be categorised as trajectory and perform a closed walk on the neighbourhood graph with the possibility of accepting a worse solution temporarily to escape local minima, the former are discontinuous and tend to jump through the search space (Birattari et al., 2001).

From Table B.2, the following conclusions can be drawn: TS, followed by SA, is the favoured single-solution search metaheuristic to approach POPs, while in general population-based metaheuristics, especially GA and PSO, are the most employed methodologies. It also becomes evident that, while the methodologies employed to approach index tracking have been applied to enhance index tracking, multi-objective portfolio optimisation has received more attention than single-objective portfolio optimisation with respect to population-based methodological coverage.

Furthermore, no single-solution approach has been used to address the multi-objective POP. It is further striking that ant colony optimisation (ACO) is by most the favoured metaheuristic to address the selection of project portfolios. Lastly it is noteworthy that hybridisation of different metaheuristics in order to improve the optimisation solutions has increased together with the complexity of the methodologies. In the following, the metaheuristic approaches to solving the individual subproblems are reviewed.

Traditional Portfolio Optimisation

While the original Markowitz problem can be solved using quadratic programming, metaheuristics have increasingly been employed to cope with the fact that the problem becomes NP-hard when more realistic constraints are introduced (Beasley, 2013). In effect, cardinality constraints, quantity constraints, and pre-assignment constraints have received overwhelming attention in the literature. The cardinality constraint defines a lower and upper limit for the numbers of assets included in the portfolio. While the lower bound aims at portfolio diversification, the upper bound accounts for the fact that marginal benefits of diversification diminish after a certain threshold (Maringer, 2005), which increases managerial efforts and transaction costs. The quantity constraint sets boundaries for the weights of included assets. While the lower limit ensures a minimum investment as smaller investments may be prohibitively costly due to transaction costs (Kolm et al., 2014), the upper limit prevents excessive exposure to a particular asset. Finally, the pre-assignment constraint enables the investor to include certain assets in the portfolio based on individual preferences independent from their risk-return characteristics.

Single-objective Portfolio Optimisation

The classical POP can be considered a single-objective optimisation problem with either one of the following model formulations: the investor minimises the risk exposure subject to a minimum attainable expected return, or the investor maximises the expected return for a given maximum level of risk. The first variant can be formulated as follows (Chang et al., 2000): A quadratic objective function is computed by aggregating over the covariances of the constituent asset returns and then minimised:

$$Min \sum_{i=1}^{N} \sum_{j=1}^{N} w_i w_j \sigma_{ij}, \tag{B.1}$$

subject to a minimum desired rate of return, the constraint that the weights have to add up to one, and the constraint that all asset weights must lie between zero and one, inclusive, thus eliminating short selling as a measure of preventing investors from excessive risk-taking by restricting them to the available budget. In formal terms:

$$\sum_{i=1}^{N} w_i \mu_i = R^*, \tag{B.2}$$

$$0 \leq w_i \leq 1, \quad \forall i = 1, 2, ..., N \tag{B.3}$$

where N is the total number of available assets, μ_i is the expected return of an asset i, R^* is the minimum required return, w are the respective weights of the assets making up the portfolio, and σ_{ij} is the covariance between two assets i and j.

Chang et al. (2000) solved the above classical problem definition using three different metaheuristic approaches (GA, SA, and TS) in order to generate a cardinality-constrained efficient frontier. They suggested pooling the results from the different approaches because no single heuristic was uniformly dominating in all observed datasets. However, Soleimani et al. (2009) introduced sector capitalisation and minimum transaction lots as further constraints and found that the GA they developed outperformed TS and SA. Following the suggestion of Chang et al. (2000) and combining GA, TS, and SA, Woodside-Oriakhi et al. (2011) explored the pooling option. They found that, on average, SA contributes little to the performance of the process and that thus a pooled GA and TS algorithm is superior to single metaheuristic approaches at the expense of higher computational time.

As for the application of strict single metaheuristic methodologies, PSO was found to be competitive with all three of the previously employed algorithms (GA, TS, and SA) for the cardinality-constrained portfolio selection problem and especially successful in low-risk portfolios (Cura,

2009). To evaluate the performance of PSO for even more realistic instances, Golmakani and Fazel (2011) further introduced minimum transaction lots, bounds on holdings, and sector capitalisation in addition to cardinality constraints. These authors applied a combination of binary PSO and improved PSO (CBIPSO), and found that CBIPSO outperforms GA in that it provides better solutions in less computing time, especially for large-scale problems. As constraints become increasingly complex, the question of constraint-handling in determining feasible solutions arises. Reid and Malan (2015) investigated this research line and developed a portfolio repair constraint handling technique applied in a PSO portfolio optimisation. Employing this, they were able to further improve the performance of the metaheuristic, again particularly for large instances.

Di Tollo and Roli (2008) provided a survey concerned with the early applications of metaheuristics to the POP and some of the proposed constraints explicitly highlighting the potential use of hybrid approaches. Likewise, such a hybrid method was proposed by Maringer and Kellerer (2003), who employed a hybrid local search algorithm combining principles of SA and evolutionary algorithms (EA) to optimise a cardinality-constrained portfolio. By combining exact mathematical programming and metaheuristic methods, Woodside-Oriakhi et al. (2011) further hybridised and created different matheuristics. This option was also investigated by Schaerf (2002) and Di Gaspero et al. (2011) who respectively combined TS and first descent (FD) and steepest descent (SD) local search metaheuristics with quadratic programming to optimise a portfolio while accounting for cardinality constraints, lower and upper boundaries for the quantity of an included asset, and pre-assignment constraints. According to their results, the developed solver finds the optimal solution in several instances and is at least comparable to other state-of-the-art methods for the others. Concerning optimality, Cesarone et al. (2013) were able to develop an exact increasing set algorithm that, for small instances, solves the POP with quantity and cardinality constraints optimally and can be extended into a heuristic procedure to account for larger instances. It outperforms the metaheuristics employed by Di Gaspero et al. (2011) and Schaerf (2002) in all instances.

Multi-objective Portfolio Optimisation

While single-objective optimisation methods consider either a minimal risk for a given expected return or a maximum return for a given expected level of risk, multi-objective optimisation methods combine two objective measures into a single one that is to be optimised (Mishra et al., 2014) or, more often, find a set of Pareto solutions while balancing two or more objective functions simultaneously. With respect to single-objective optimisation methods that require the *ex-ante* definition of an acceptable degree of profitability, multi-objective optimisation requires no previous knowledge about the investor's degree of risk aversion and is thus a more general approach transferrable to different decision-makers. The approach of combining risk and return characteristics into a single objective function is taken by Zhu et al. (2011). They introduced the Sharpe ratio as a simultaneous measure and, since GA and PSO have been found to be competitively successful in solving the single-objective version, performed a comparison of these metaheuristics in solving the non-linear constrained portfolio optimisation problem. As previously established, they also argue that PSO outperforms GAs, especially in large instances. While they did not include realistic constraints other than a total portfolio weight equal to one in addition to portfolio assets restricted to positive weights, in which the short selling of the portfolio's underlying assets is prohibited, the authors also investigated unrestricted portfolios. The solution portfolios obtained with the PSO solver outperformed those constructed using GA for all test problems in terms of Sharpe ratio, and the established efficient frontier was above that of GA portfolios in all but one instance.

According to Streichert et al. (2003), the multi-objective POP can be formulated employing two simultaneous objective functions as follows. For a multi-objective optimisation it becomes necessary to minimise the portfolio risk expressed by the portfolio variance:

$$Min \sum_{i=1}^{N} \sum_{j=1}^{N} w_i w_j \sigma_{ij}, \tag{B.4}$$

while maximising the return of the portfolio, i.e.:

$$Max \sum_{i=1}^{N} w_i \mu_i, \tag{B.5}$$

subject to:

$$\sum_{i=1}^{N} w_i = 1, \tag{B.6}$$

$$0 \leq w_i \leq 1, \quad \forall i = 1, 2, ..., N. \tag{B.7}$$

Alternatively, equations B.4 and B.5 can be combined into a single one by incorporating objective weights as follows (Mishra et al., 2014):

$$Min\ \lambda \sum_{i=1}^{N} \sum_{j=1}^{N} w_i w_j \sigma_{ij} - (1 - \lambda) \sum_{i=1}^{N} w_i \mu_i, \tag{B.8}$$

subject to the aforementioned constraints. In this case, the weights as determined by the parameter λ represent the risk aversion of the investor. By varying this parameter and running repeatedly, a Pareto efficient frontier can be established. Because of the high performance of PSO in solving the single-objective POP, enhanced PSO algorithms for solving the multi-objective POP have been proposed by Deng et al. (2012) and He and Huang (2012). Cardinality and bounding constraints were incorporated by Deng et al. (2012) who find that their algorithm mostly outperforms GA, SA, and TS algorithms as well as previous PSO approaches, especially in the case of low-risk portfolios. It can be concluded that different findings unanimously favour PSO in situations when low-risk investment is demanded in addition to a larger-scale potential asset pool. Similarly, He and Huang (2012) proposed a modified PSO (MPSO) algorithm that outperforms regular PSO for their four optimisation sets. More recently, they also developed a new PSO to deal with discontinuous modelling of the POP and find that it generally outperforms PSO and also performs better than MPSO in larger search spaces (He and Huang, 2014). Other population-based algorithms applied in optimising cardinality-constrained portfolios include firefly algorithms (FA) (Tuba and Bacanin, 2014b) and artificial bee colony (ABC) algorithms (Tuba and Bacanin, 2014a). However, because the results were satisfactory at most even after modifications, the authors hybridised FA and ABC by incorporating the FA search strategy into ABC to enhance exploitation and found that their data suggested superiority of the methodology compared to GA, SA, TS, and PSO (Tuba and Bacanin, 2014a). Streichert et al. (2003) accounted for further constraints: buy-in thresholds (acquisition prices) and round lots (smallest volume of an asset that can be purchased). They employed two multi-objective evolutionary algorithms (MOEA): GA and an EA enhanced through the integration of a local search that applies Lamarckism, thus allowing the individual improvements to be passed on to the offspring. They found that this enhancement greatly improved the reliability of the results, especially with respect to the additional constraints. Unfortunately, these approaches are hardly reproducible due to their complexity, reinforcing the need for a less metaphorical and more scientifically reproducible approach.

Nevertheless, apart from the neglect of realistic non-linear constraints, there is a second point of criticism to the original Markowitz model, namely its assumption of normal financial returns, which, in reality are characterised by a leptokurtic distribution (Krink and Paterlini, 2011), making it necessary to consider non-parametric risk measures. Such a measure is the value-at-risk, as employed by Babaei et al. (2015) who developed two multi-objective algorithms based on PSO to solve a cardinality- and quantity-constrained POP. Through splitting the whole swarm into sub-swarms that are then evolved distinctly, their methodology outperformed similar benchmark metaheuristics. In order to optimise a non-parametric value-at-risk and to include further constraints, including lower and upper bounds for the weights of included assets, a threshold for asset weight changes, lower and upper bounds for the weights of one asset class and a turnover rate that determines the maximum asset allocation changes possible at once, Krink and Paterlini (2011) developed the differential evolution (DE) for multi-objective portfolio optimisation (DEMPO) algorithm. An extended version of a generalised DE metaheuristic was also employed in optimising a highly constrained POP by Ayodele and Charles (2015). The included constraints consist of bounds on holdings, cardinality, minimum transaction lots, and expert opinion. An expert can form an opinion based on indicators beyond the scope of the analysed data and influence whether or not an asset should be included. Their methodology showed improved performance when compared to GA, TS, SA, and PSO. Lwin et al. (2014) considered cardinality, quantity, preassignment and round lot constraints and developed a multi-objective evolutionary algorithm that

is improved through a learning-guided solution generation strategy, which promotes efficient convergence (learning-guided multi-objective evolutionary algorithm with external archive, MODE-wAwL). It was shown that the developed algorithm outperformed four benchmark state-of-the-art multi-objective evolutionary algorithms in that its efficient frontier was superior.

An extensive review of the application of evolutionary algorithms to the POP is provided by Metaxiotis and Liagkouras (2012). Likewise, for an extensive review on different portfolio optimisation problems, including single- and multi-objective optimisation, the reader is referred to Mansini et al. (2014). It can be asserted that population-based metaheuristics have yielded superior results compared to single-solution metaheuristics in the case of single-objective portfolio optimisation. This has resulted in them being dominantly applied to multi-objective portfolio optimisation.

Passive Investment

Closely related to portfolio optimisation as an active portfolio management strategy, passive investment strategies have received less attention in the optimisation literature. These strategies are characterised by limited on-going buying and selling, as well as by ensuing limited maintenance. Based on the traditional capital market theory stating that market portfolios offer the greatest return per unit of risk, passive investment strategies have been shown to outperform actively managed funds and thus gained popularity (Alexander and Dimitriu, 2004).

Index Tracking

The index tracking problem (ITP) is a passive portfolio management strategy in that investors aim at mimicking a market or sector index. This is done by either replicating the index or by selecting a portfolio that follows the index behaviour as closely as possible without including all the stocks that make up the original index. In the case of perfect replication, there are transaction costs associated with updating the portfolio to continuously accurately depict the index, which thus have to be deducted when evaluating the performance. Therefore, the ITP is largely concerned with the latter, partial replication. There are thus two stages in index tracking, the common goal of which is to minimise the resulting tracking error (the distance between the portfolio and benchmark returns). The first consists of selecting the assets to include in the portfolio and the second relates to determining the weights. Thus, it consists of a combinatorial and a continuous numerical problem, which both have to be addressed simultaneously (Krink et al., 2009). Once similar constraints as in portfolio optimisation are introduced (e.g. floor and ceiling constraints, cardinality constraints, pre-assignments, or class constraints), minimising the objective function of the tracking error becomes extraordinarily difficult to solve with exact methods.

The optimisation problem can thus be addressed with the following formulation (Beasley et al., 2003). Minimise the tracking error:

$$Min\ E = \frac{\left[\sum_{t \in S} |r_t - R_t|^{\alpha}\right]^{\left(\frac{1}{\alpha}\right)}}{T}, \tag{B.9}$$

where $S = 1, 2, ..., T$ are the time periods considered during which the portfolio return was below that of the tracked index, r_t is the tracking portfolio return, R_t is the return of the tracked index itself, and α is the penalisation power that is applied to the difference between the realised return and the benchmark return. If we set $\alpha = 2$, the tracking error is defined as the root mean square error (RMSE). In the case of a perfect reproduction of an index, the tracking error would naturally be equal to zero. In the most basic formulation, the following constraints have to be considered:

$$\sum_{i=1}^{N} z_i = K, \tag{B.10}$$

which represents the cardinality constraint and ensures that any new tracking portfolio contains K stocks, as z_i takes on the value of one if a stock is included in the replication portfolio and zero otherwise. As in portfolio optimisation, the weights have to be limited:

$$0 < w_i \leq 1,\ z_i = 1,\ \ \forall i = 1, 2, ..., N. \tag{B.11}$$

This limits the weights of the included stocks to be larger than zero and equal to or below one. The non-included stocks must naturally dispose of a weighting of zero:

$$w_i = 0,\ z_i = 0,\ \ \forall i = 1, 2, ..., N \tag{B.12}$$

Maringer and Oyewumi (2007) investigated partial replication and introduced cardinality constraints concerning upper and lower weight limits and integer constraints in the ITP employing a DE methodology. Their findings suggest that partial replication is indeed sufficient in replicating the benchmark index. This is due to the fact that only a decreasing marginal improvement is reached by increasing the cardinality.

Scozzari et al. (2013) were able to develop a mixed integer quadratic programming formulation to solve the ITP including hard constraints set by the European Union on ceilings of asset inclusion weights as well as low turnover rates and resulting low transaction costs in small instances. However, the introduction of realistic constraints generally makes it difficult to use exact methods in solving large ITP instances. Early research by Beasley et al. (2003) introduced a population-based evolutionary metaheuristics to solve the partial reproduction ITP with regard to stock indices including constraints on transaction costs (as well as a ceiling for the total inclusion of stocks). Derigs and Nickel (2004) developed a two-stage SA metaheuristic, in which they controlled for cardinality constraints and transaction costs through turnover volume restrictions.

For larger instances, especially in multi-period analysis, Scozzari et al. (2013) proposed hybridising metaheuristics with exact methods. This has been done by Krink et al. (2009) who addressed the two subtasks of ITP simultaneously and applied a metaheuristic approach based on DE combined with a combinatorial search operator. Although their developed methodology initially failed to find acceptable solutions, they showed that extending DE with a search operator by selecting the assets with highest weights in the benchmark improved the results greatly in comparison with GA, SA, and PSO. Ruiz-Torrubiano and Suárez (2009) employed a GA hybridised with quadratic programming. More recently, Ni and Wang (2013) also tackled the ITP employing a hybridised GA with increased learning ability that is enabled through goal programming. The authors included cardinality and integer constraints, as well as proportion constraints for individual portfolio assets. While both methodologies yielded successful solutions, the models neglect transaction costs, which are however indicated by the authors as a variable important to investigate in future research. The trade-off between transaction costs and tracking performance was then investigated by Chiam et al. (2013) who developed a multi-objective evolutionary index tracking platform that considers multiple periods and simultaneously optimises tracking performance and transaction costs while considering round lots and non-negativity constraints as well as floor constraints as buy-in threshold to prevent unnecessary transaction costs and capital injections.

Although different metaheuristic approaches have been chosen to cope with the realistic constraints of the ITP, Affolter et al. (2016) found that due to the missing measure to define the distance between portfolios with respect to their assets and weights, invasive weed optimisation (IWO) did not lead to satisfactory optimisation results. Di Tollo and Maringer (2009) created a framework for classifying the metaheuristics applied to ITP and present a review of the literature.

Enhanced Index Tracking Beasley et al. (2003) defined an objective function that accounts for a trade-off between the tracking error and excess returns above those of the benchmark index. This enhanced index tracking allows the manager discretion in pursuing risk-limited active strategies to enhance return. Considering that investors might see a trade-off between the trading error and excess returns above the index has led to the enhanced index tracking problem (EITP), in which investors aim at beating the benchmark index. This can either be done through active selection of the included assets and weights or through a passive extension of the methodology by incorporating the excess return as a further optimisation objective. The EITP then becomes a multi-objective optimisation problem, in which the tracking error is minimised while maximising the degree of beating the benchmark index so that a solution dominates another if the excess return is higher given the same level of trading inaccuracy or if the trading accuracy for the same level of excess return exceeds that of the other solution. This can be formulated by including a second objective function that defines the excess return between r_t and R_t:

$$Min\ E = \frac{\left[\sum_{t \in S} |r_t - R_t|^{\alpha}\right]^{\left(\frac{1}{\alpha}\right)}}{T}, \tag{B.13}$$

while maximising the excess return r^*:

$$Max\, r^* = \sum_{t=1}^{T} \frac{r_t - R_t}{T}, \tag{B.14}$$

subject to the aforementioned constraints:

$$\sum_{i=1}^{N} z_i = K, \tag{B.15}$$

$$0 \leq w_i \leq 1, z_i = 1, \quad \forall i = 1, 2, ..., N, \tag{B.16}$$

$$w_i = 0, z_i = 0, \quad \forall i = 1, 2, ..., N. \tag{B.17}$$

Canakgoz and Beasley (2009) solved the ITP as well as the EITP including transaction costs, an upper limit on the total number of stocks purchased, and a limit on the incurred transaction costs using exact methods (mixed-integer linear programming formulations). However, Li et al. (2011b) showed they could mostly outperform the methodology employed by Canakgoz and Beasley (2009) by implementing an immunity-based optimisation algorithm. It is an EA based on the clonal selection of an immune system, or the immune response to antigens (De Castro and Von Zuben, 2002). Including further constraints, Li and Bao (2014) also employed an immunity-based multi-objective optimisation algorithm with non-negativity and floor and ceiling buy-in thresholds. They concluded that the inclusion of optimisation of the tracking process in addition to optimising tracking error and excess return is valuable as the optimisation of the tracking process improves results in most instances. A perfectly enhanced tracking portfolio would outperform the index by a low-frequency trend such as steady excess return while negative returns should be trendless and characterised by high frequency variation. Thus, the tracking process can be enhanced by considering different frequencies for tracking error and excess returns when the former is minimised and the latter maximised (Li and Bao, 2014). Optimisation of the tracking process is expected to increase in importance for multi-period assessment; the authors, however, leave this for further research. The question of multi-periodicity was investigated by Andriosopoulos et al. (2013) who addressed the EITP employing both DE and GA. They could show that the so-constructed mimicking portfolios inhibit less risk compared to the underlying benchmark index, while proficiently replicating their performance. Nevertheless, they concluded that the GA version outperforms DE in terms of minimum tracking errors, as well as maximum mean excess returns. As they explicitly considered different time horizons for rebalancing the portfolio, these authors reinforced the idea that there exists a trade-off between transaction costs, which decrease with longer rebalancing periods, and Sharpe ratios (as a measure of the tracking performance and profitability), which is negatively impacted by decreased rebalancing frequency as investigated by Chiam et al. (2013) for the ITP.

An alternative approach was pursued by Guastaroba and Speranza (2012) who applied a kernel search framework to both the ITP and the EITP. They argued that error measurements should be undertaken as absolute values and introduced the possibility that an investor already holds a portfolio as a further constraint to consider in addition to transaction costs. However, they treated the EITP as a single-objective optimisation by outperforming the market index, while keeping the tracking error below a given threshold. Compared to a general-purpose solver, the performance of the kernel search model was superior. Further including metaheuristics into the optimisation, Thomaidis (2011) considered an EITP problem with restrictions on the maximum of tradable assets, and employed fuzzy set theory to consider non-standard investment objectives, such as the probability of under-performing. The resulting cardinality-constrained problem was solved using nature-inspired optimisation techniques: SA, GA, and PSO.

Lastly, while some authors declare active and passive portfolio management as mutually exclusive concepts, the close connection between index tracking and portfolio optimisation could be illustrated by the approach taken by Di Tollo et al. (2014) who combined the two methods in a multi-criteria optimisation problem. They employed a hybrid metaheuristic consisting of local search metaheuristics (FD, SD and TS) and quadratic programming to estimate the efficient frontier. Combining the concepts of risk and return with tracking error led to a three-dimensional

objective function and Pareto frontiers. The developed methodology was found competitive in performance with other metaheuristics such as TS.

Project Portfolio Selection

Unlike banks and institutional investors, non-financial companies as well as governments are faced with a different type of portfolio choice. As a method to determine which proposals to pursue and the corresponding budget allocation, investment or project appraisal is related to portfolio optimisation in its goal of maximising a benefit figure. This figure can be monetary, but also related to knowledge gain in the case of research projects. Usually, decisions cannot be altered or adjusted during the course of the projects, or at least not without incurring considerable financial losses. Thus, investment appraisal determines a strategic organisational path for the medium and long term. This problem becomes NP-hard due to its sheer complexity (Fernandez et al., 2015). It is by its very nature a multi-period problem and the budget-allocating entity usually pursues several conflicting objectives, some of which can be of qualitative nature. For that matter, Doerner et al. (2004) proposed a two-stage procedure. During the first phase, the Pareto frontier is constructed. Then, in the second phase, it is interactively explored by the decision-makers to account for personal preferences. The optimisation process is carried out in the first phase. A formal description of this problem, based on the one presented in Doerner et al. (2004), is included next. The benefit function $b_{l,t}(x)$ that comprises the value of the l different benefit groups, such as generated funds, cash flows, patents or other beneficial outcomes of the selected projects is to be maximised over all considered time periods t for all included projects, i.e.:

$$b_{l,t}(x) = \sum_{i=1}^{N} b_{i,l,t} x_i, \tag{B.18}$$

where x_i is a binary variable that takes on the value of one for included projects and zero otherwise, subject to constraints concerning resource limitations $R_{q,t}$ that apply to all resource categories r_q, such as budget, capacity, or manpower, as well as minimum benefit requirements $B_{l,t}$ that define a threshold below which the decision-maker is uninterested in the implementation of projects:

$$r_{q,t}(x) \leq R_{q,t}; q = 1, ..., R \; and \; t = 1, 2, ..., T, \tag{B.19}$$

$$b_{l,t}(x) \leq B_{l,t}; l = 1, ..., B \; and \; t = 1, 2, ..., T. \tag{B.20}$$

Because of the modelled similarities, the methodological approaches employed are inspired by the research on traditional portfolio optimisation. Early work (Ghasemzadeh and Archer, 2000) conducted optimisation after the construction of a weighted objective function and constraints concerning budget and man-hours in an integer linear programming approach. However, test instances were very limited because the authors aspired a comparison between manually computed portfolios and those constructed employing their decision support system. For their metaheuristic two-stage approach Doerner et al. (2004) employed Pareto ACO (P-ACO). As there are possible synergies between projects that should be evaluated in order to accurately estimate the benefits of a project portfolio, the authors made an attempt at incorporating these considerations into their methodology and pointed out that, unlike GA, SA, and TS that are adaptive metaheuristics, P-ACO specifically constructs project portfolios through pheromone vectors. This has two advantages. Firstly, infeasible solutions are avoided and secondly, project interactions can more naturally be considered in the construction of solutions. They further took into account floor and ceiling constraints for inclusion of projects from any given subset, as well as resource limitations and minimum benefit requirements for individual projects. Compared to Pareto SA and a non-dominated sorting GA (NSGA), P-ACO yielded the most efficient results. This approach was then further enhanced by Stummer and Sun (2005), who compared the performance of a P-ACO procedure enhanced through adding a neighbourhood search routine, a TS procedure, and a variable neighbourhood procedure. Their findings suggested that the improved P-ACO model performs better than TS with many objective functions and a large set of efficient solutions and is thus specifically suitable for real-life problems. Furthermore, Doerner et al. (2006) concluded that including both a

learning and a two step integer linear pre-processing procedure to initialise several initial efficient project portfolios improves performance of the P-ACO algorithm.

More recently, research has also drawn on findings from other areas, such as scheduling: Gutjahr et al. (2008) and Gutjahr et al. (2010) also took employee competencies and the evolution of their knowledge scores over time through learning or depreciation into account. While the earlier work optimised a weighted average objective function using ACO and GA metaheuristic procedures and found the GA to be superior when the search space is not highly constrained, the authors developed a multi-objective optimisation model, which simultaneously optimises the objectives of maximum economic gains and aggregated competence increase in their later work. They also divided the problem into master and slave subproblems, the first of which is concerned with the project selection, while the slave problem optimises the allocation of personnel to the projects over time. Although the slave problem can be solved using exact methods, the master problem was solved using the NSGA-II and P-ACO metaheuristics. While both performed reasonably well, NSGA-II outperformed P-ACO in synthetic test instances, while P-ACO outperformed NSGA-II for the investigated real-life instances. Carazo et al. (2010) further investigated this research line and included scheduling as a continuative concept following the project selection. Their developed metaheuristics approach is based on scatter search (SS) for project portfolio selection (SS-PPS). As previous work, they also considered interdependences between different projects and can show that their model outperformed other heuristic approaches based on EA (SPEA). Similar to Rabbani et al. (2010), who presented a multi-objective PSO metaheuristic and found it to be competitive with respect to SPEA II, Urli and Terrien (2010) formulated the project portfolio selection problem as a multi-objective non-linear integer program, which they solved using the SSPMO metaheuristic (Molina et al., 2007). In a first phase, they generated an initial set of efficient solutions through TS and then combined these via SS. While this approach solved small and medium instances in satisfactory computation time, the determination of all non-dominated project portfolios still remains difficult when considering large, but realistically relevant instances (100 projects or more). While this might not be relevant in most firm investment decisions, it is a significant drawback for governments or bodies awarding funding for projects.

Another issue that has only recently been addressed is project divisibility. While business projects are at least partially indivisible, research projects funded by governments can often also be executed with partial funding and it is thus a further question how much of the sought after funding is awarded, introducing further constraints to the budget allocation. Hence, more recent research increasingly focused on large-scale instances and partial allocation. Cruz et al. (2014) used ACO in solving a stationary project portfolio optimisation problem, in which partial support of the requested budget was allowed. They developed a non-outranked ACO approach, incorporating a fuzzy outranking preference model. Unlike previous research, they assumed that the preferences of the decision-maker are to some extent known. Outranking was employed in an *a priori* preference system in order to model that decision-makers will have preferences towards different portfolios on the efficient frontier based on their personal goals concerning the achievement of objectives. Incorporating these preferences allows identifying those portfolios that lie on the efficient frontier and simultaneously are not outranked by another portfolio. They incorporated budgetary constraints in that they defined upper and lower bounds for inclusion of projects from a particular group. Fernandez et al. (2015) further enhanced this approach by including integer linear programming methods to generate an initial population and thus hybridising the metaheuristic further. They also included synergies in their optimisation, concluding that their model outperformed state-of-the-art metaheuristics. It can be asserted that project synergies, project divisibility, the incorporation of multi-periodicity, and outranking are the prominent real-life constraints and trends that specifically increase the complexity of the portfolio selection process and thus distinguish this COP from a classical POP.

5. Risk Management

Risk management of financial and non-financial companies refers to the evaluation, often in real time, of realistic data concerning the institution's exposure to a certain source of risk and it is further concerned with statistics on trends that will influence that exposure in the future. While quantitative data is relevant and necessary for this, it must be complemented by qualitative information for informed decision-making, both in financial as well as non-financial institutions (Chorafas, 2007). Risk management is addressed in terms of optimisation through metaheuristics

for credit risk assessment and the resulting bankruptcy prediction. García et al. (2015) provide a detailed review of developed systems and, to a less obvious extent, applications to the optimisation of trading rules in the financial markets. As depicted previously for portfolio management, Table B.3 presents the metaheuristic methodologies applied to the different subproblems of risk management. As before, the numbers inside each cell refer to the number of articles reviewed for each topic and methodology.

TABLE B.3: The application of traditional metaheuristics and hybridisation to subproblems of risk management

Optimisation Problem	Single-solution search		Population-based search										Hybrid
	SA	TS	GA	ACO	EA	ABC	PSO	SS	HBMO	FA	BA	HS	
Credit risk assessment		2	4	1			1	1	1				6
Bankruptcy prediction			4		1		2						6
Optimisation in stock trading	2		4			1	2			1	1	1	7
Optimisation in foreign exchange trading			3		1								4

From Table B.3, several conclusions can be drawn. Firstly, GA are the preferred metaheuristics in risk management as well; their popularity is expressed through the use in every single reviewed problem. Furthermore, PSO has also received widespread attention. Contrary to that, more exotic algorithms, such as harmony search (HS), firefly algorithms (FA), or bat algorithms (BA). Secondly, it can be seen that bankruptcy prediction – as an advancement of credit risk analysis –, as well as optimisation of trading systems for foreign exchange markets – as an advancement of optimisation in stock trading –, have received less attention in the literature and have been approached with fewer methodologies. They thus represent interesting future research lines. Thirdly, it becomes evident that hybridisation among metahueristics or other optimisation methods is far more prevailing in risk management optimisation than in portfolio optimisation. Lastly, it is evident that relatively recently developed metaheuristics, such as IWO and honeybees mating optimisation (HBMO), have not been applied as comprehensively as well-established ones.

Credit Risk Assessment and Optimisation

Credit risk assessment is one of the most researched and recognised topics in the banking industry. There are many different approaches and sophisticated credit risk assessment tools for financial institutions. However, during the last years, non-financial companies have also recognised the need to treat their trade credits to customers with the same caution and scrutiny. Thus, both financial and non-financial analysts have to decide on the granting as well as the extension of loans. While the use of metaheuristics is still scarce in this area of application, they are increasingly used as a pre-processing procedure in order to identify the most relevant predictors of credit risk in the analysis of large datasets of information. Marinakis et al. (2008) classified a set of companies into different classes of credit risk level. They propose and compare TS, GA, and ACO for solving the feature selection subset problem, which are then used in determining the appropriate level of credit risk. The employed accuracy measures are determined by whether or not a subject has been classified in the right category. In a simple two-classes model, this is based on the four scenarios depicted in Table B.4.

TABLE B.4: Definitions of the classified and the misclassified samples

		Actual class	
		1	2
Estimated class	1	T_1	F_2
	2	F_1	T_2

The overall classification accuracy (OCA) can then serve as optimisation objective that is to be maximised:

$$Max\ OCA = \frac{T_1 + T_2}{T_1 + F_1 + T_2 + F_2} * 100. \tag{B.21}$$

This framework can be extended to include as many different credit risk classes as the decision-maker considers.

More recently, Marinaki et al. (2010) employed HBMO in determining the relevant features. They were able to show that this metaheuristic reduced the number of used features by more than half and still yielded superior results compared to PSO, ACO, GA, and TS. Oreski et al. (2012) employed neural networks (NN) hybridised with GA (GA-NN) to enhance the classification accuracy of the NN classifiers by choosing optimal features. They found that the prediction ability was as accurate as in the case of using all available data features in the analysis. In a subsequent work, Oreski and Oreski (2014) further improved the results by employing a hybrid GA instead of GA. Their results suggested that they hence achieved a higher and less volatile accuracy with on average fewer selected features through a reduction of the search space and an incremental phase of the GA. Chi and Hsu (2012) employed GA in selecting relevant variables to combine a bank's internal behavioural scoring model with an external credit bureau scoring model and thus creating a dual scoring model that subsequently outperformed the individual model in credit risk prediction. A survey on the importance of employing the right fitness function in the GA for credit assessment is provided in Kozeny (2015).

Trends in credit risk assessment concern the hybridisation of metaheuristics with other techniques for feature selection. Wang et al. (2010) developed a feature selection based on rough set and TS (FSRT). In comparison with non-preselecting models, the savings in computational time and performance accuracy were significant. Similarly, Wang et al. (2012) used a rough set and scatter search feature selection (RSFS) that is able to improve results in all three considered base sets, i.e. neural network model, J48 decision tree and logistic regression (LR). Lastly, Danenas and Garsva (2015) pursued the idea of optimising the classifiers of a linear support vector machine (SVM) using PSO. While their results were comparable to the use of other classifiers (LR and RBF networks), the proposed methodology, however, delivered less stable performance.

Bankruptcy Prediction

Closely related to credit risk evaluation is the prediction of bankruptcy of firms. Strictly defined, bankruptcy occurs when debtors are unable to repay outstanding debts. While bankruptcy prediction constitutes part of the credit risk evaluation process, it is vital for banks and companies to constantly monitor their credit risk exposure. Because of the two-classes framework (firms that go bankrupt and firms that do not), the basic optimisation framework is similar as suggested for credit risk assessment. The difficulty and difference to credit risk assessment lies however in the relatively longer aspired forecasting period and the difficulty in predicting the exact time of bankruptcy.

TABLE B.5: Definitions of the classified and the misclassified samples for bankruptcy prediction

		Actual class	
		Bankrupt	Not bankrupt
Estimated class	Bankrupt	T_1	F_2
	Not bankrupt	F_1	T_2

$$Max\ OCA = \frac{T_1 + T_2}{T_1 + F_1 + T_2 + F_2} * 100. \tag{B.22}$$

Especially in practical bankruptcy prediction, it is worth considering to differently value the two classes of mistakes that occur. While falsely classifying a subject as bankruptcy candidate (type II misclassification) merely leads to missed revenues, a false classification as healthy company (type I misclassification) usually leads to at least partial failure on a payment and thus has greater consequences for profitability. This is thus an improvement to the methodology open for further research.

Early research conducted by Back et al. (1996) highlighted the contribution of GA in predicting bankruptcy when hybridised with artificial neural networks (ANN). Shin and Lee (2002) introduced the prediction of corporate bankruptcy using GA and historical financial data. Kim and Han (2003) further employed GA to extract decision rules based on qualitative expert decisions

and find their methodology to be superior compared to neural networks or inductive learning methods because the rules created by GA are more accurate and have larger coverage. An extensive survey on the early research in this knowledge area can be found in Kumar and Ravi (2007) who reviewed both statistical and computing methods. Their evaluation concluded that all statistical methods are outperformed by the most popular neural network methodology, back propagation neural networks. They further highlighted the prediction accuracy of SVM and pushed for further research in the area of hybridising different soft computing approaches. More recently, Kirkos (2015) presented the literature on artificial intelligence and machine learning techniques employed in bankruptcy prediction.

Following the conclusions of Kumar and Ravi (2007) another research line attempts to optimise SVM with metaheuristics. Min et al. (2006) improved the performance of SVM with regards to optimising the feature subset and parameters simultaneously. They showed that selecting an appropriate feature subject has implications for the kernel and that their integration improved bankruptcy prediction accuracy. Chen (2011) highlighted that while intelligent techniques provide higher prediction accuracy for smaller datasets and are adversely affected by increasing datasets, statistical methods perform more accurately when the dataset is large. But the author also indicated that a hybrid between PSO and SVM could yield a good balance between short- and long-term prediction accuracy. This was consequently done by Lu et al. (2015) who combined switching PSO (SPSO) and SVM. The SPSO was employed in searching the optimal parameter values of radial basis function (RBF) kernel of the SVM and the authors showed that this hybridisation yielded superior results to GA-SVM and PSO, respectively. These findings were supported by Chen (2011) and Chen (2014a) who also employed PSO-SVM and showed high accuracy with a significantly reduced number of parameters. Furthermore, Gaspar-Cunha et al. (2014) proposed an evolutionary multi-objective approach that simultaneously minimises the number of features and maximises the accuracy of the classifier in SVM so that the algorithm is self-adaptive. The general advantage of multi-objective optimisation lies in the attainment of a set of efficient solutions from which the decision-maker can perform a trade-off based on personal preferences.

Recently, ensemble learning has been applied to the bankruptcy prediction problem (BPP). Kim and Kang (2010) proposed hybridising an ensemble with neural networks and showed that it improved prediction accuracy compared to regular neural networks. However, these attempts often suffer from multicollinearity, or high correlation among the individual classifiers, and thus Kim and Kang (2012) improved their methodology to include a GA-based coverage optimisation to alleviate multicollinearity through classifier selection. More recently, Davalos et al. (2014) developed an accurate GA-based ensemble classifier model with heterogeneous instead of individual classifiers that is comprehensible due to its if-then-structure. They showed that the fitness function can be tailored to accommodate further constraints and showed the improved performance of their approach.

However, the financial ratios employed in the main research lines are unavailable for a large portion of companies. Small and medium-sized enterprises (SMEs) do not dispose of regular audited financial data or market prices and public ratings due to publicly traded equity or debt instruments and it is necessary to include available and relevant indicators for these individual firms. Thus, with special regards to SMEs, Gordini (2014) compared the prediction accuracy of GA, SVM, and LR. The author showed that the prediction of GA was superior, especially with regard to type II misclassifications (assuming bankruptcy when this is not the case) and with regard to prediction of bankruptcy for small firms. This is especially relevant because financial data for small firms is often only incompletely available and thus impedes the bankruptcy analysis. Furthermore, the reduction of type II misclassification improves business relationships between SMEs and prevents reputational damage, which might lead to credit crunches, especially in small firms.

Optimisation of Decision-Support Systems for Trading

As a result of the above optimisations in predicting markets and prices, the development and optimisation of automated trading systems has become of prevalent importance and special interest for broker investment banks and other institutional investors alike. As for forecasting, a large portion of the literature addresses stock trading, while some researchers have concentrated on the foreign exchange markets.

Stock Market Trading Derigs and Nickel (2003) developed a decision support system (DSS) for portfolio optimisation and index tracking that can be tailored to meet the constraints and portfolio types of different institutional agents. They stressed the importance of hard (government-imposed and compulsory) and soft (shaped by preferences of the investor) constraints. They implemented a local search guided by SA in order to optimise the DSS with respect to floor and ceiling constraints and transaction costs. These authors have shown for the application to passive tracking of the DAX 30 that their developed system delivered solutions with minimal tracking errors in acceptable computing time.

Focusing on real-time decisions, Chavarnakul and Enke (2009) proposed a trading system for the stock market based on volume adjusted moving average (VAMA) that is hybridised with neural networks to decrease the time of receiving trading signals (which is of major importance for the adoption of a real-time trading framework), fuzzy logic to deal with uncertainty, and GA techniques to optimise the trading signals to overall increase efficiency. Depending on the strength and direction of a given signal, the system assumes a buy or sell position. If the signal is not confident enough, a hold position is taken. The so established neuro-fuzzy based GA (NF-GA) was shown to lead to fewer trades and thus reduced transaction costs, while profitability was increased.

Gorgulho et al. (2011) also proposed a system to automatically manage a portfolio of assets and highlighted the necessity of adapting the system to the state of the market to optimise performance. They employed GA and technical analysis rules. The system adapts to the different states of the market and always outperforms the random and at most instances the buy and hold strategy. The system requires the user to input the available budget, the maximum of assets to be included in the portfolio at any time, whether or not short selling is considered and the allowed amount of transaction costs. Then, an initial portfolio is constructed, as it would be in the POP, employing a GA. However, over the course of the investment, the system regularly updates the proposal based on technical trading rules based on closing positions that are either prone to losses or have achieved a profit and can be closed and refilling empty positions. Teixeira and De Oliveira (2010) combined technical trading rules with nearest neighbour classification. Their analysis was solely based on historical data of stock closing prices and volume, on the basis of which trading rules were formed. Their proposed system outperforms a buy and hold strategy in most cases in terms of profitability. Because however, the parameters in these functions have to be determined, metaheuristics have been applied in the optimisation. The hybridisation of technical trading rules and metaheuristics is seen as an especially promising research area. Brasileiro et al. (2013) further refined the strategy by Teixeira and De Oliveira (2010) by searching for the best system parameters (the parameter of the classifier and the values of the stop loss and stop gain) and number of lags with an ABC algorithm and thus outperforming the previous trading system as well as the buy and hold strategy in most instances. Nunez-Letamendia (2007) had already shown that GA is robust and powerful when applied to optimising technical trading rules. Similarly, Lin et al. (2011) applied a GA to improve trading rules for individual stocks, which are then combined with echo state networks to provide suggestions for trading. Their results showed an outperformance of the buy and hold strategy as well as significant profits even in bear market situations. In a more recent work, Wang et al. (2014a) employed a time-variant PSO (TVPSO) to determine the optimal parameter values of a complex trading system: performance-based reward strategy (PRS). PRS combines moving average and trading range breakout trading rules, which are combined based on weights that are determined by previous performance. Considering transaction costs and excluding short selling, the system was able to achieve high profits and the application of the TVPSO significantly optimised the trading system.

As previously applied to bankruptcy prediction and credit risk assessment, hybridisations of metaheuristics and ANN have also recently shown to provide accurate forecasts of stock market prices. While both provide better results than a passive buy and hold strategy, harmony search (HS) based models have been shown to outperform GA based models with regards to forecasting errors (Göçken et al., 2016). Very recently, the hybridisation of data mining techniques with metahueristics – e.g., firefly algorithms (FA), bat algorithms (BA), and PSO – has created clustering metaheuristics able to predict patterns in the general movement of stock markets, such as periods of lower and higher return (Prasanna and Ezhilmaran, 2015). These insights together with automated trading systems could further enhance investment decisions.

Foreign Exchange Market Trading

Foreign exchange market trading can either concern hedging foreign exchange risk or speculation. Only the latter has the objective of making a profit by exploiting market inefficiencies and is thus the central assumption in trading systems. Speculation in the foreign exchange markets requires the application of real-time trading systems to an even greater extent than stock market trading due to the nature of trading profits, which are realised at the time of trading. Myszkowski and Bicz (2010) established a trading system based on decision trees that consider technical trading indicators. EA then generates trading strategies. While these were able to achieve a profit, the system is still too fragile to be used in automated trading, but can serve as additional indicator to investors. Mendes et al. (2012) proposed employing GA to optimising trading rules and although their developed trading system performs well in terms of computational time because of the small population size employed, it fails to perform well in terms of profitability when faced with transaction costs. Zhang and Ren (2010) developed a high-frequency trading system based on the optimisation of technical indicators through GA that was able to produce annualised profits. In addition to intraday prediction, highlighting the importance of real-time, Evans et al. (2013) implemented a trading system based on feed forward neural networks with back propagation architecture, whose topology was optimised using GA. In comparison with Zhang and Ren (2010), they were able to considerably improve annualised net profits.

6. Linkage Between Portfolio Optimisation and Risk Management

Despite the fact they have been addressed as two independent types of problems in most of the scientific literature, this section highlights the relationship between portfolio selection and risk management. In the first place, portfolio optimisation problems directly consider a risk measure (such as portfolio variance, portfolio semivariance, value at risk, alpha and beta, among others) and, therefore, they can also be seen as risk management problems. For example, the specification of adequate risk measures that accurately depict return distributions is a well-established area of research of on-going interest within academia and especially financial institutions. It is directly concerned with one-dimensional risk measures of individual assets and multi-dimensional measure to account for interaction of asset portfolios (Rachev et al., 2010), unambiguously linking risk management and portfolio optimisation.

In the second place, most risk management models related to optimisation problems can be seen as rich variants of portfolio optimisation problems. For instance, stock market trading is in the essence of the problems concerned with constructing an initial portfolio and updating it over time to reflect current macro- and microeconomic developments. Likewise, while credit risk and bankruptcy risk are only estimated in the overwhelming majority of the risk management literature, it is the ultimate goal to build low-risk portfolios (e.g. loan or customer portfolios) by including preferably those assets with a lower credit risk and excluding other assets expected to go bankrupt. Using a multi-objective EA, Moreno-Paredes et al. (2013) explicitly acknowledge the linkage between credit risk management and portfolio optimisation by treating the loan decision among a set of customers as a credit portfolio optimisation problem (CPOP). More generally, implicit in a portfolio optimisation problem is pooling assets with imperfectly correlated returns that leads to a diversification of idiosyncratic sources of risk and a reduction in the overall risk of portfolio investment.

Fig. B.5 depicts the relationship between risk management and portfolio optimisation problems reviewed in this paper. They have been divided according to whether single- or multi-objective optimisation approaches have been taken in solving them. The extension of the ovals representing risk management problems and portfolio optimisation problems respectively signifies the extension of possible solving approaches beyond traditional optimisation techniques. The intersecting set comprises the CPOP, as well as stock trading. The risk management problems of bankruptcy and credit risk prediction are located directly on the border to portfolio optimisation, as the predictions are generally employed in a following portfolio selection process. Foreign exchange trading, unlike stock trading, is considered a sole risk management problem. While stock trading consists of the establishment and maintaining of a portfolio, speculative profits in foreign exchange trading are generated through simultaneous buying and selling and not the establishment of a portfolio.

Concerning the subproblems of portfolio optimisation, the ITP and EITP do not directly consider risk measures. Unlike that, the POP is directly concerned with risk minimisation and thus closely related to risk management problems.

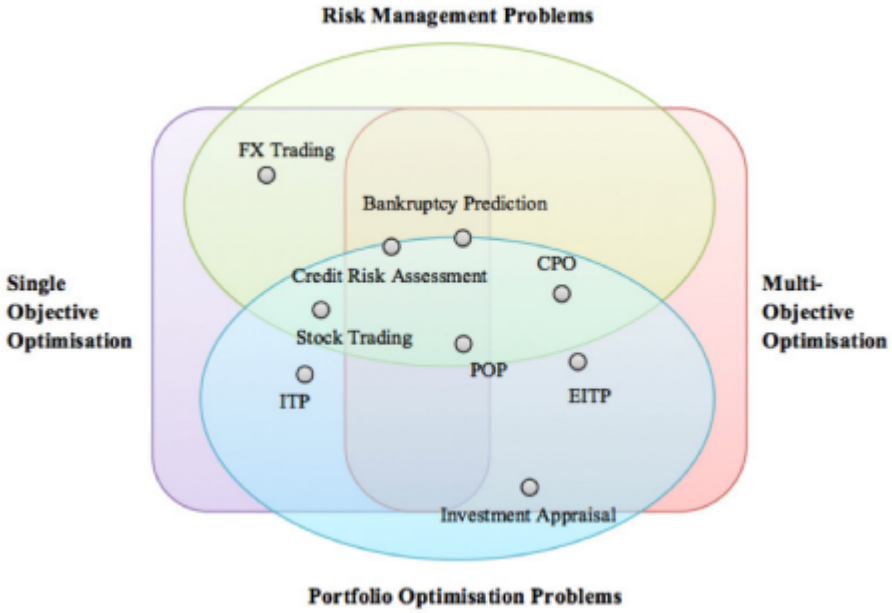


FIGURE B.5: A Unified Classification of Portfolio Optimisation and Risk Management

7. Emerging Trends in the Literature

From the previous discussion, one clear trend that we see in the literature is the transfer of methodological knowledge from portfolio optimisation to risk management optimisation problems. In effect, since very efficient metaheuristics have been developed to solve single- and multi-objective POPs, and since most optimisation problems in the risk management arena can be seen as enriched variants of POPs, it is reasonable to assume that these metaheuristics will constitute the base for developing new solving approaches in the risk management field. Another trend is related to the increasing complexity of the problems being addressed, which makes even more evident the need for faster (or parallelizable) metaheuristic approaches. These 'fast metaheuristics' will be needed as the models introduce further constraints to account for real-life circumstances, and as the real-time factor that is required in most of the decision-making processes will become even more relevant. Strongly related to this point, distributed and parallel computing techniques can be employed to accelerate the 'wall clock times' necessary to obtain near-optimal solutions to large-scale problems (Juan et al., 2013b). Furthermore, the fact that two or more objectives have to be considered simultaneously to account for the complexity has shown to increase the employment of multi-objective optimisation techniques.

Another clear trend that can be derived from the previous analysis of the literature is the predominance in the use of population-based metaheuristics over single-point metaheuristics. It is our opinion that no family of metaheuristics are shown to be superior in performance (regarding both quality of solutions as well as computing times) to another. At least, we have not found any scientific evidence that supports that claim. Thus, a lot of research can be done yet regarding the use of single-point metaheuristics (other than SA and TS) in solving both rich variants of portfolio optimisation problems and risk management optimisation problems. Related to this, it is possible to observe in the reviewed literature a clear trend to develop hybrid algorithms, which combine different types of metaheuristics as well as metaheuristics with machine learning and statistical techniques. While hybridisation can be an effective strategy to solve complex problems, it might also add some degree of additional complexity to the solving algorithms. This, in turn, might make them more difficult to be clearly understood, correctly implemented, and applied in practical scenarios. Adding complexity to algorithms –e.g., in the form of additional parameters that require fine tuning– also makes it difficult to reproduce their experimental results by independent

researchers. For those reasons, it is our opinion that only in cases in which significant improvements in performance are obtained (both in solutions quality and computing times), is the hybridisation of algorithms justified. As it has happened before in other application fields, we expect the emergence of relatively easy-to-implement and understand metaheuristics (either single-solution or population-based ones with a few number of parameters) that can be competitive and flexible (i.e., adaptable to different variants of the problem without many effort).

With regards to data, recent research has shown a trend to employ different risk measures to more accurately depict the characteristics of the underlying data. This is also a particularly interesting research area as further stakeholders of financial optimisations (e.g. SMEs) do not provide traditional optimisation inputs (financial data) and thus alternative measurements of risk are promising. Further concerning measuring, while hybridisations of simulation and optimisation have recently been developed and gained popularity in the application to stochastic combinatorial optimisation problems in different application areas (Juan et al., 2015a), the above finance-related problems have not yet been extensively addressed by simheuristics, even though financial data is characterised by stochastic macro- as well as firm-level uncertainty. It can thus be expected that this research line is promising, with respect to both, the design of new problems at the interface of the two main research areas that have been treated separately in the literature but are fairly interrelated and the application of simheuristics to established COPs that previously neglected stochastic uncertainty on the one hand and the newly formulated COPs on the other.

8. Conclusions

In this paper we have reviewed the state-of-the-art regarding the use of metaheuristics in the fields of portfolio optimisation problems and risk management problems. From this review, several conclusions can be extracted: *(i)* the number of related publications has been increasing during the last decade, especially in the case of portfolio optimisation problems; *(ii)* population-based metaheuristics, and in particular GA and PSO, have been the predominant solving methodologies; *(iii)* regarding single-solution metaheuristics, TS and SA have been extensively applied too; *(iv)* there is not a 'single winner' approach, meaning that different metaheuristic implementations have provided results of comparable quality to different problems; *(v)* there is a clear trend in promoting the development of hybrid algorithms, either by combining different metaheuristics or by combining metaheuristics with machine/statistical learning techniques; *(vi)* most portfolio optimisation problems include some kind of risk management and, in the other direction, most risk management problems considering optimisation issues can be modelled as enriched variants of portfolio optimisation problems; and *(vii)* there is a lack of works considering stochastic versions of the optimisation problems, i.e. random variables modelling inputs such as return rates, or risk measurements, but also uncertainty in the objective function, or constraints.

From this analysis of the state-of-the-art, we foresee different open research lines that need to be fully explored by researchers and practitioners, among others: *(i)* methodological knowledge transfer between the more studied portfolio optimisation problem and risk management optimisation problems; *(ii)* development of easy-to-implement and fast metaheuristics (e.g., variable neighbourhood search, iterated local search, etc.) that require few parameters and perform similar to more complex ones; *(iii)* hybridisation of metaheuristics with machine/statistical learning methods to account for dynamic behaviour in time-evolving systems; *(iv)* hybridisation of metaheuristics with simulation (simheuristics) to account for uncertainty; and *(v)* use of distributed and parallel computing paradigms to allow for 'real-time' solutions in complex decision-making processes.

Acknowledgements

This work has been partially supported with doctoral grants from the Open University of Catalonia, the Spanish Ministry of Economy and Competitiveness and Feder (TRA2013-48180-C3-3-P, TRA2015-71883-REDT). Likewise we want to acknowledge the support received by the Department of Universities, Research & Information Society of the Catalan Government (Grant 2014-CTP-00001).

References

Affolter, K., T. Hanne, D. Schweizer, and R. Dornberger (2016). "Invasive weed optimization for solving index tracking problems". In: *Soft Computing* 20.9, pp. 3393–3401.

Alexander, C. and A. Dimitriu (2004). "Equity indexing: Optimize your passive investments". In: *Quantitative Finance* 4.3, pp. 30–33.

Andriosopoulos, K., M. Doumpos, N. C. Papapostolou, and P. K. Pouliasis (2013). "Portfolio optimization and index tracking for the shipping stock and freight markets using evolutionary algorithms". In: *Transportation Research Part E: Logistics and Transportation Review* 52, pp. 16–34.

Asta, S. (2015). "Machine learning for improving heuristic optimisation". PhD thesis. The University of Nottingham, UK.

Ayodele, A. A. and K. A. Charles (2015). "Portfolio selection problem using generalized differential evolution 3". In: *Applied Mathematical Sciences* 9.42, pp. 2069–2082.

Babaei, S., M. M. Sepehri, and E. Babaei (2015). "Multi-objective portfolio optimization considering the dependence structure of asset returns". In: *European Journal of Operational Research* 244.2, pp. 525–539.

Back, B., T. Laitinen, and K. Sere (1996). "Neural networks and genetic algorithms for bankruptcy predictions". In: *Expert Systems with Applications* 11.4, pp. 407–413.

Beasley, J. E. (2013). "Portfolio optimisation: Models and solution approaches". In: *Tutorials in operations research* 10, pp. 201–221.

Beasley, J. E., N. Meade, and T.-J. Chang (2003). "An evolutionary heuristic for the index tracking problem". In: *European Journal of Operational Research* 148.3, pp. 621–643.

Birattari, M., L. Paquete, T. Stützle, and K. Varrentrapp (2001). *Classification of metaheuristics and design of experiments for the analysis of components*. Tech. rep. Techn. Rep. N. AIDA-01-05, Techn. Univ. Darmstadt.

Boussaïd, I., J. Lepagnot, and P. Siarry (2013). "A survey on optimization metaheuristics". In: *Information Sciences* 237, pp. 82–117.

Brasileiro, R. C., V. L. F. Souza, B. J. T. Fernandes, and A. L. I. Oliveira (2013). "Automatic method for stock trading combining technical analysis and the artificial bee colony algorithm". In: *2013 IEEE Congress on Evolutionary Computation*, pp. 1810–1817.

Canakgoz, N. A. and J. E. Beasley (2009). "Mixed-integer programming approaches for index tracking and enhanced indexation". In: *European Journal of Operational Research* 196.1, pp. 384–399.

Carazo, A. F., T. Gómez, J. Molina, A. G. Hernández-Díaz, F. M. Guerrero, and R. Caballero (2010). "Solving a comprehensive model for multiobjective project portfolio selection". In: *Computers & operations research* 37.4, pp. 630–639.

Cesarone, F., A. Scozzari, and F. Tardella (2013). "A new method for mean-variance portfolio optimization with cardinality constraints". In: *Annals of Operations Research* 205.1, pp. 213–234.

Chang, T.-J., N. Meade, J. E. Beasley, and Y. M. Sharaiha (2000). "Heuristics for cardinality constrained portfolio optimisation". In: *Computers & Operations Research* 27.13, pp. 1271–1302.

Chavarnakul, T. and D. Enke (2009). "A hybrid stock trading system for intelligent technical analysis-based equivolume charting". In: *Neurocomputing* 72.16, pp. 3517–3528.

Chen, M.-Y. (2011). "A hybrid model for business failure prediction-utilization of particle swarm optimization and support vector machines". In: *Neural Network World* 21.2, p. 129.

— (2014a). "Using a hybrid evolution approach to forecast financial failures for Taiwan-listed companies". In: *Quantitative Finance* 14.6, pp. 1047–1058.

Chi, B.-W. and C.-C. Hsu (2012). "A hybrid approach to integrate genetic algorithm into dual scoring model in enhancing the performance of credit scoring model". In: *Expert Systems with Applications* 39.3, pp. 2650–2661.

Chiam, S. C., K. C. Tan, and A. Al Mamun (2013). "Dynamic index tracking via multi-objective evolutionary algorithm". In: *Applied Soft Computing* 13.7, pp. 3392–3408.

Chorafas, D. N. (2007). "What is meant by risk management?" In: *Risk Management Technology in Financial Services*, pp. 24–42.

Cruz, L., E. Fernandez, C. Gomez, G. Rivera, and F. Perez (2014). "Many-objective portfolio optimization of interdependent projects with 'a priori'incorporation of decision-maker preferences". In: *Appl. Math* 8.4, pp. 1517–1531.

Cura, T. (2009). "Particle swarm optimization approach to portfolio optimization". In: *Nonlinear Analysis: Real World Applications* 10.4, pp. 2396–2406.

Danenas, P. and G. Garsva (2015). "Selection of support vector machines based classifiers for credit risk domain". In: *Expert Systems with Applications* 42.6, pp. 3194–3204.

Davalos, S., F. Leng, E. H. Feroz, and Z. Cao (2014). "Designing an if–then rules-based ensemble of heterogeneous bankruptcy classifiers: A genetic algorithm approach". In: *Intelligent Systems in Accounting, Finance and Management* 21.3, pp. 129–153.

De Castro, L. N. and F. J. Von Zuben (2002). "Learning and optimization using the clonal selection principle". In: *IEEE transactions on evolutionary computation* 6.3, pp. 239–251.

Deng, G.-F., W.-T. Lin, and C.-C. Lo (2012). "Markowitz-based portfolio selection with cardinality constraints using improved particle swarm optimization". In: *Expert Systems with Applications* 39.4, pp. 4558–4566.

Derigs, U. and N. Nickel (2003). "Meta-heuristic based decision support for portfolio optimization with a case study on tracking error minimization in passive portfolio management". In: *OR Spectrum* 25.3, pp. 345–378.

Derigs, U. and N. H. Nickel (2004). "On a local-search heuristic for a class of tracking error minimization problems in portfolio management". In: *Annals of Operations Research* 131.1-4, pp. 45–77.

Di Gaspero, L., G. Di Tollo, A. Roli, and A. Schaerf (2011). "Hybrid metaheuristics for constrained portfolio selection problems". In: *Quantitative Finance* 11.10, pp. 1473–1487.

Di Tollo, G. and D. Maringer (2009). "Metaheuristics for the index tracking problem". In: *Metaheuristics in the Service Industry*. Springer, pp. 127–154.

Di Tollo, G. and A. Roli (2008). "Metaheuristics for the portfolio selection problem". In: *International Journal of Operations Research* 5.1, pp. 13–35.

Di Tollo, G., T. Stützle, and M. Birattari (2014). "A metaheuristic multi-criteria optimisation approach to portfolio selection". In: *Journal of Applied Operational Research* 6.4, pp. 222–242.

Doerner, K., W. J. Gutjahr, R. F. Hartl, C. Strauss, and C. Stummer (2004). "Pareto ant colony optimization: A metaheuristic approach to multiobjective portfolio selection". In: *Annals of operations research* 131.1-4, pp. 79–99.

Doerner, K. F., W. J. Gutjahr, R. F. Hartl, C. Strauss, and C. Stummer (2006). "Pareto ant colony optimization with ILP preprocessing in multiobjective project portfolio selection". In: *European Journal of Operational Research* 171.3, pp. 830–841.

Dorigo, M. (1992). "Optimization, learning and natural algorithms". PhD thesis. Politecnico di Milano, Italy.

Evans, C., K. Pappas, and F. Xhafa (2013). "Utilizing artificial neural networks and genetic algorithms to build an algo-trading model for intra-day foreign exchange speculation". In: *Mathematical and Computer Modelling* 58.5, pp. 1249–1266.

Farmer, J. D., N. H. Packard, and A. S. Perelson (1986). "The immune system, adaptation, and machine learning". In: *Phys. D* 2.1-3, pp. 187–204.

Feo, T. A. and M. G. C. Resende (1989). "A probabilistic heuristic for a computationally difficult set covering problem". In: *Oper. Res. Lett.* 8.2, pp. 67–71.

Feo, T. A and M. G. C. Resende (1995). "Greedy randomized adaptive search procedures". In: *Journal of global optimization* 6.2, pp. 109–133.

Fernandez, E., C. Gomez, G. Rivera, and L. Cruz-Reyes (2015). "Hybrid metaheuristic approach for handling many objectives and decisions on partial support in project portfolio optimisation". In: *Information Sciences* 315, pp. 102–122.

García, V., A. I. Marqués, and J. S. Sánchez (2015). "An insight into the experimental design for credit risk and corporate bankruptcy prediction systems". In: *Journal of Intelligent Information Systems* 44.1, pp. 159–189.

Gaspar-Cunha, A., G. Recio, L. Costa, and C. Estébanez (2014). "Self-adaptive MOEA feature selection for classification of bankruptcy prediction data". In: *The Scientific World Journal* 2014.

Gendreau, M. and J.-Y. Potvin (2010). *Handbook of metaheuristics*. 2nd. Springer Publishing Company, Incorporated.

Ghasemzadeh, F. and N. P. Archer (2000). "Project portfolio selection through decision support". In: *Decision support systems* 29.1, pp. 73–88.

Gilli, M. and E. Schumann (2012). "Heuristic optimisation in financial modelling". In: *Annals of operations research* 193.1, pp. 129–158.

Gilli, M., D. Maringer, and E. Schumann (2011). *Numerical methods and optimization in finance*. Academic Press.

Glover, F. (1977). "Heuristics for integer programming using surrogate constraints". In: *Decision Science* 8.1, pp. 156–166.

— (1986). "Future paths for integer programming and links to artificial intelligence". In: *Computers & Operations Research* 13.5, pp. 533–549.

Göçken, M., M. Özçalıcı, A. Boru, and A. T. Dosdoğru (2016). "Integrating metaheuristics and artificial neural networks for improved stock price prediction". In: *Expert Systems with Applications* 44, pp. 320–331.

Golmakani, H. R. and M. Fazel (2011). "Constrained portfolio selection using particle swarm optimization". In: *Expert Systems with Applications* 38.7, pp. 8327–8335.

Gordini, N. (2014). "A genetic algorithm approach for SMEs bankruptcy prediction: Empirical evidence from Italy". In: *Expert Systems with Applications* 41.14, pp. 6433–6445.

Gorgulho, A., R. Neves, and N. Horta (2011). "Applying a GA kernel on optimizing technical analysis rules for stock picking and portfolio composition". In: *Expert systems with Applications* 38.11, pp. 14072–14085.

Guastaroba, G. and M. G. Speranza (2012). "Kernel search: An application to the index tracking problem". In: *European Journal of Operational Research* 217.1, pp. 54–68.

Gutjahr, W. J., S. Katzensteiner, P. Reiter, C. Stummer, and M. Denk (2008). "Competence-driven project portfolio selection, scheduling and staff assignment". In: *Central European Journal of Operations Research* 16.3, pp. 281–306.

Gutjahr, W. J., S. Katzensteiner, C. Reiter P.and Stummer, and M. Denk (2010). "Multi-objective decision analysis for competence-oriented project portfolio selection". In: *European Journal of Operational Research* 205.3, pp. 670–679.

He, G. and N. Huang (2012). "A modified particle swarm optimization algorithm with applications". In: *Applied Mathematics and Computation* 219.3, pp. 1053–1060.

— (2014). "A new particle swarm optimization algorithm with an application". In: *Applied Mathematics and Computation* 232, pp. 521–528.

Holland, J. H. (1962). "The compact genetic algorithm". In: *Journal of the ACM* 3.9, pp. 297–314.

Juan, A. A., J. Faulin, J. Jorba, J. Caceres, and J. M. Marques (2013b). "Using parallel & distributed computing for real-time solving of vehicle routing problems with stochastic demands". In: *Annals of Operations Research* 207.1, pp. 43–65.

Juan, A. A., J. Faulin, S. Grasman, M. Rabe, and G. Figueira (2015a). "A review of simheuristics: extending metaheuristics to deal with stochastic optimization problems". In: *Operations Research Perspectives* 2, pp. 62–72.

Kennedy, J. and R. C. Eberhart (1995). "Particle swarm optimization". In: *Proceedings of the IEEE International Conference on Neural Networks*, pp. 1942–1948.

Kim, M.-J. and I. Han (2003). "The discovery of experts' decision rules from qualitative bankruptcy data using genetic algorithms". In: *Expert Systems with Applications* 25.4, pp. 637–646.

Kim, M.-J. and D.-K. Kang (2010). "Ensemble with neural networks for bankruptcy prediction". In: *Expert Systems with Applications* 37.4, pp. 3373–3379.

— (2012). "Classifiers selection in ensembles using genetic algorithms for bankruptcy prediction". In: *Expert Systems with applications* 39.10, pp. 9308–9314.

Kirkos, E. (2015). "Assessing methodologies for intelligent bankruptcy prediction". In: *Artificial Intelligence Review* 43.1, pp. 83–123.

Kolm, P. N., R. Tütüncü, and F. J. Fabozzi (2014). "60 years of portfolio optimization: Practical challenges and current trends". In: *European Journal of Operational Research* 234.2, pp. 356–371.

Kozeny, V. (2015). "Genetic algorithms for credit scoring: Alternative fitness function performance comparison". In: *Expert Systems with Applications* 42.6, pp. 2998–3004.

Krink, T. and S. Paterlini (2011). "Multiobjective optimization using differential evolution for real-world portfolio optimization". In: *Computational Management Science* 8.1-2, pp. 157–179.

Krink, T., S. Mittnik, and S. Paterlini (2009). "Differential evolution and combinatorial search for constrained index-tracking". In: *Annals of Operations Research* 172.1, pp. 153–176.

Kumar, P. R. and V. Ravi (2007). "Bankruptcy prediction in banks and firms via statistical and intelligent techniques–A review". In: *European journal of operational research* 180.1, pp. 1–28.

Li, Q. and L. Bao (2014). "Enhanced index tracking with multiple time-scale analysis". In: *Economic Modelling* 39, pp. 282–292.

Li, Q., L. Sun, and L. Bao (2011b). "Enhanced index tracking based on multi-objective immune algorithm". In: *Expert Systems with Applications* 38.5, pp. 6101–6106.

Lin, X., Z. Yang, and Y. Song (2011). "Intelligent stock trading system based on improved technical analysis and echo state network". In: *Expert systems with Applications* 38.9, pp. 11347–11354.

Lu, Y., N. Zeng, X. Liu, and S. Yi (2015). "A new hybrid algorithm for bankruptcy prediction using switching particle swarm optimization and support vector machines". In: *Discrete Dynamics in Nature and Society* 2015.

Lwin, K., R. Qu, and G. Kendall (2014). "A learning-guided multi-objective evolutionary algorithm for constrained portfolio optimization". In: *Applied Soft Computing* 24, pp. 757–772.

Malkiel, B. G. (2003). "Passive investment strategies and efficient markets". In: *European Financial Management* 9.1, pp. 1–10.

Mansini, R., W. Ogryczak, and M. G. Speranza (2014). "Twenty years of linear programming based portfolio optimization". In: *European Journal of Operational Research* 234.2, pp. 518–535.

Marinaki, M., Y. Marinakis, and C. Zopounidis (2010). "Honey bees mating optimization algorithm for financial classification problems". In: *Applied Soft Computing* 10.3, pp. 806–812.

Marinakis, Y., M. Marinaki, and N. Matsatsinis (2008). "A stochastic nature inspired metaheuristic for clustering analysis". In: *International Journal of Business Intelligence and Data Mining* 3.1, pp. 30–44.

Maringer, D. (2005). *Portfolio management with heuristic optimization*. Berlin: Springer.

Maringer, D. and H. Kellerer (2003). "Optimization of cardinality constrained portfolios with a hybrid local search algorithm". In: *OR Spectrum* 25.4, pp. 481–495.

Maringer, D. and O. Oyewumi (2007). "Index tracking with constrained portfolios". In: *Intelligent Systems in Accounting, Finance and Management* 15.1-2, pp. 57–71.

Markowitz, H. (1952). "Portfolio selection". In: *The journal of finance* 7.1, pp. 77–91.

Mendes, L., P. Godinho, and J. Dias (2012). "A Forex trading system based on a genetic algorithm". In: *Journal of Heuristics* 18.4, pp. 627–656.

Metaxiotis, K. and K. Liagkouras (2012). "Multiobjective evolutionary algorithms for portfolio management: a comprehensive literature review". In: *Expert Systems with Applications* 39.14, pp. 11685–11698.

Michalewicz, Z. and D. B. Fogel (2013). *How to solve it: Modern heuristics*. Springer Science & Business Media.

Min, S.-H., J. Lee, and I. Han (2006). "Hybrid genetic algorithms and support vector machines for bankruptcy prediction". In: *Expert systems with applications* 31.3, pp. 652–660.

Mishra, S. K., G. Panda, and R. Majhi (2014). "Constrained portfolio asset selection using multiobjective bacteria foraging optimization". In: *Operational Research* 14.1, pp. 113–145.

Mladenovic, N. (1995). "A variable neighborhood algorithm: A new metaheuristic for combinatorial optimization". In: *Abstracts of papers presented at Optimization Days*, p. 112.

Molina, J., M. Laguna, Rafael. Martí, and R. Caballero (2007). "SSPMO: A scatter tabu search procedure for non-linear multiobjective optimization". In: *INFORMS Journal on Computing* 19.1, pp. 91–100.

Moreno-Paredes, J. C., C. Mues, and L. C. Thomas (2013). "Credit scoring and credit control XIII Conference Proceedings". In: chap. A multi-objective decision framework for credit portfolio management.

Myszkowski, P. B. and A. Bicz (2010). "Evolutionary algorithm in Forex trade strategy generation". In: *Computer Science and Information Technology (IMCSIT), Proceedings of the 2010 International Multiconference on*, pp. 81–88.

Ni, H. and Y. Wang (2013). "Stock index tracking by Pareto efficient genetic algorithm". In: *Applied Soft Computing* 13.12, pp. 4519–4535.

Nunez-Letamendia, L. (2007). "Fitting the control parameters of a genetic algorithm: An application to technical trading systems design". In: *European journal of operational research* 179.3, pp. 847–868.

Oreski, S. and G. Oreski (2014). "Genetic algorithm-based heuristic for feature selection in credit risk assessment". In: *Expert systems with applications* 41.4, pp. 2052–2064.

Oreski, S., D. Oreski, and G. Oreski (2012). "Hybrid system with genetic algorithm and artificial neural networks and its application to retail credit risk assessment". In: *Expert systems with applications* 39.16, pp. 12605–12617.

Osman, I. H. and J. P. Kelly (1996). "Meta-heuristics: An overview". In: *Meta-heuristics: Theory & Applications*. Kluwer Academic, pp. 1–22.

Patrick, H. T. (1966). "Financial development and economic growth in underdeveloped countries". In: *Economic Development and Cultural Change* 14.2, pp. 174–189.

Pearl, J. (1984). "Heuristics: intelligent search strategies for computer problem solving". In: *Addison-Wesley Pub. Co., Inc., Reading, MA*.

Prasanna, S. and D. Ezhilmaran (2015). "Stock market prediction using clustering with meta-heuristic approaches". In: *Gazi University Journal of Science* 28.3, pp. 395–403.

Rabbani, M., M. A. Bajestani, and G. B. Khoshkhou (2010). "A multi-objective particle swarm optimization for project selection problem". In: *Expert Systems with Applications* 37.1, pp. 315–321.

Rachev, S. T., B. Racheva-Iotova, S. V. Stoyanov, and F. J. Fabozzi (2010). "Risk management and portfolio optimization for volatile markets". In: *Handbook of Portfolio Construction*, pp. 493–508.

Reid, S. G. and K. M. Malan (2015). "Constraint handling methods for portfolio optimization using particle swarm optimization". In: *Computational Intelligence, 2015 IEEE Symposium Series on*, pp. 1766–1773.

Ruiz-Torrubiano, R. and A. Suárez (2009). "A hybrid optimization approach to index tracking". In: *Annals of Operations Research* 166.1, pp. 57–71.

Schaerf, A. (2002). "Local search techniques for constrained portfolio selection problems". In: *Computational Economics* 20.3, pp. 177–190.

Scozzari, A., F. Tardella, S. Paterlini, and T. Krink (2013). "Exact and heuristic approaches for the index tracking problem with UCITS constraints". In: *Annals of Operations Research* 205.1, pp. 235–250.

Shin, K.-S. and Y.-J. Lee (2002). "A genetic algorithm application in bankruptcy prediction modeling". In: *Expert Systems with Applications* 23.3, pp. 321–328.

Soleimani, H., H. R. Golmakani, and M. H. Salimi (2009). "Markowitz-based portfolio selection with minimum transaction lots, cardinality constraints and regarding sector capitalization using genetic algorithm". In: *Expert Systems with Applications* 36.3, pp. 5058–5063.

Sörensen, K. (2015). "Metaheuristics—the metaphor exposed". In: *International Transactions in Operational Research* 22.1, pp. 3–18.

Streichert, F., H. Ulmer, and A. Zell (2003). "Evolutionary algorithms and the cardinality constrained portfolio selection problem". In: *Operations Research Proceedings 2003, Selected Papers of the International Conference on Operations Research (OR 2003)*.

Stummer, C. and M. Sun (2005). "New multiobjective metaheuristic solution procedures for capital investment planning". In: *Journal of Heuristics* 11.3, pp. 183–199.

Talbi, E.-G. (2009). *Metaheuristics: From design to implementation*. New Jersey: John Wiley & Sons.

Teixeira, L. A. and A. L. I. De Oliveira (2010). "A method for automatic stock trading combining technical analysis and nearest neighbor classification". In: *Expert systems with applications* 37.10, pp. 6885–6890.

Thomaidis, N. S. (2011). "A soft computing approach to enhanced indexation". In: *Natural Computing in Computational Finance*. Springer, pp. 61–77.

Tuba, M. and N. Bacanin (2014a). "Artificial bee colony algorithm hybridized with firefly algorithm for cardinality constrained mean-variance portfolio selection problem". In: *Appl. Math. Inf. Sci* 8.6, pp. 2831–2844.

— (2014b). "Upgraded firefly algorithm for portfolio optimization problem". In: *Computer Modelling and Simulation (UKSim), 2014 UKSim-AMSS 16th International Conference on*. IEEE, pp. 113–118.

Urli, B. and F. Terrien (2010). “Project portfolio selection model, a realistic approach”. In: *International Transactions in Operational Research* 17.6, pp. 809–826.

Wang, F., L. H. Philip, and D. W. Cheung (2014a). “Combining technical trading rules using particle swarm optimization”. In: *Expert Systems with Applications* 41.6, pp. 3016–3026.

Wang, J., K. Guo, and S. Wang (2010). “Rough set and tabu search based feature selection for credit scoring”. In: *Procedia Computer Science* 1.1, pp. 2425–2432.

Wang, J., A.-R. Hedar, S. Wang, and J. Ma (2012). “Rough set and scatter search metaheuristic based feature selection for credit scoring”. In: *Expert Systems with Applications* 39.6, pp. 6123–6128.

Woodside-Oriakhi, M., C. Lucas, and J. E. Beasley (2011). “Heuristic algorithms for the cardinality constrained efficient frontier”. In: *European Journal of Operational Research* 213.3, pp. 538–550.

Zhang, H. and R. Ren (2010). “High frequency foreign exchange trading strategies based on genetic algorithms”. In: *Networks Security Wireless Communications and Trusted Computing (NSWCTC), 2010 Second International Conference on*. Vol. 2, pp. 426–429.

Zhu, H., Y. Wang, K. Wang, and Y. Chen (2011). “Particle swarm optimization for the constrained portfolio optimization problem”. In: *Expert Systems with Applications* 38.8, pp. 10161–10169.

B.5 Rich portfolio optimization with stocks and individual commodity futures contracts

Jana Doering[1], Laura Calvet[2], Renatas Kizys[3], Àngels Fitó[1], Angel A. Juan[2]

1. Economics and Business Department, Open University of Catalonia – IN3,
Av. Tibidabo 39-43, 08035 Barcelona, Spain
Tel.: +34-603-894448
e-mail: {jdoering, afitob}@uoc.edu
2. Computer Science Dept., Universitat Oberta de Catalunya – IN3,
Av. Carl Friedrich Gauss 5, 08060 Castelldefels, Spain
e-mail: {lcalvetl, ajuanp}@uoc.edu
3. Subject Group of Economics and Finance, Portsmouth Business School, University of Portsmouth,
Richmond Building, Portland Street, Portsmouth PO1 3DE, Hampshire, UK
e-mail: renatas.kizys@port.ac.uk

Abstract

Considering the increasing integration of international stock markets, diversification benefits of stock portfolios have become limited. Commodity futures, however, present a possible source of successful diversification due to their low correlation with stock markets. Thus, the introduction of futures into a traditional stock portfolio can be modeled as a portfolio optimization problem (POP). The goal of the POP is to minimize risk for an expected portfolio return by determining the right proportions of included assets. As additional realistic constraints are considered, the problem becomes NP-hard. Thus, metaheuristic algorithms are commonly employed for solving large instances of rich POP versions. To the best of our knowledge, there is a significant gap in the literature as the diversification benefits of stocks and futures portfolios have not yet been evaluated using metaheuristics. This paper seeks to close this gap by introducing a matheuristic algorithm which combines an iterated local search with quadratic optimization. This algorithm is then employed to determine and compare stock-alone versus stock-and-futures portfolios. The benefit of including futures is quantified, and it is shown that especially risk-averse investors can achieve further diversification by employing our approach. Furthermore, we discuss the stability of the portfolios in an ex-post analysis. The results show that risk-averse stock-and-futures portfolios are more promising in achieving actual returns exceeding the minimum required return than traditional stock-alone portfolios.

Keywords: Portfolio optimization, Commodity futures, Matheuristic, Metaheuristic, Iterated Local Search.

1. Introduction

Financial decisions are directly linked to wealth creation through capital accumulation, sustainable economic development and thus an increase in welfare (Patrick, 1966). This thriving for improvement suggests a mentality of optimization in financial decision-making. Since Markowitz' pioneering work on mean-variance optimization of portfolios (Markowitz, 1952), the research community and investors alike have shown extensive interest in advancements in modern portfolio theory, which is based on two constituting assumptions. The investor, being concerned with both the risk and the return of his investment, diversifies by incorporating imperfectly correlated assets into a portfolio. Originally, Markowitz describes the investor's decision in the portfolio optimization problem (POP) as a strategy of: *(i)* the selection of financial assets to be included in the portfolio; and *(ii)* the determination of the respective weights of the included assets. These two tasks are performed with the aim of minimizing the portfolio risk as expressed through the weighted covariances of the constituent assets subject to a minimal expected return sought by the investor, resulting in a quadratic objective function. While the problem can be solved in satisfactory time for small problem instances employing exact methods, it becomes NP-hard when realistic constraints are introduced (Beasley, 2013). Hence, an extensive adaptation of the original Markowitz formulation has taken place and in practice, this means that metaheuristics are required to solve large-size instances of realistic POPs in reasonably low computing times. Thus,

the literature has been divided into two strands. On the one hand, researchers have simplified the model to solve it using exact methods, while on the other hand, researchers have proposed new methodologies to cope with the complexity. The former approach neglects the real-life intricacies that the decision-making process confronts the investor with and the obtained solutions are expected to be of little benefit in practice. Thus, further optimization techniques, including heuristic and metaheuristic algorithms, have been implemented to address realistic and complex combinatorial optimization problems (COPs) in general and the POP in particular. While the former are problem-dependent and thus tailormade, the latter are general-purpose search methodologies that can be easily adapted to solve a range of COPs (Talbi, 2009). Metaheuristics have therefore been increasingly applied as solving approaches for real-life formulations of the POP that consider realistic constraints (Chang et al., 2000; Maringer and Kellerer, 2003; Cura, 2009; Soleimani et al., 2009; Golmakani and Fazel, 2011; Woodside-Oriakhi et al., 2011). However, while the application of metaheuristics has shown to be successful for stock portfolios, research concerning portfolios composed of multiple types of individual financial assets is heavily limited. Thus, the combination of real-life constraints and differing financial assets and the inherent trade-off between satisfying constraints and achieving highest possible diversification benefits has not been evaluated. This paper hence addresses this gap in the literature and employs a metaheuristics algorithm, more formally, an adaptive matheuristics composed of an iterated local search metaheuristic and a quadratic solver, to address a rich version of the portfolio optimization problem, considering an extensive set of realistic constraints and individual futures contracts in addition to stocks.

Recently, stock markets have become more integrated, resulting in higher positive correlation among individual stocks and thus diminishing successful diversification (You and Daigler, 2012). Because most research on metaheuristics applied to POPs relies on pre-established benchmarks, the outcomes of such a development on the quality of the established portfolios cannot be detected. To remedy the negative implications of high positive correlations between individual stocks it would be convenient to include a second asset class to exemplify possible diversification benefits. While many securities qualify as portfolio hedge, in particular, we increase diversification by considering individual commodity futures contracts because they have been found to have low correlations with stocks (Jensen et al., 2002; Chong and Miffre, 2010). Their correlational properties have been found to be caused among others by an opposite reaction of futures to macroeconomic shocks (Silvennoinen and Thorp, 2013; Bansal et al., 2014). It has thus been shown that the obtained portfolios including individual futures contracts significantly outperformed those composed of only stocks (Geman and Kharoubi, 2008; You and Daigler, 2012; Bansal et al., 2014). Accordingly, it is thus the aim of this paper and its first contribution to apply a metaheuristic solving approach to a POP that considers not only individual stocks, but also individual commodity futures. Secondly, our contribution is to quantify the benefit of doing so and thirdly to show how further diversification can be achieved. Finally, we further evaluate whether the best-found portfolios provide stability in terms of achieved short-term returns.

The paper is organized in the following way. Section 2 presents a review of the relevant literature on the application of metaheuristics to portfolio optimization, as well as on the diversification benefits of futures. Section 3 provides a formal description of the problem, while the solving methodology is presented in Section 4. In Section 5 the time series data is analysed and processed. The computational experiments and the results thereof are laid out in Section 6. Finally, Section 7 concludes and presents lines of interest for future research.

2. Literature Review

In this section, we review the two strands of literature that we aim to merge in this paper. In particular, we review how previous research indicates in what ways commodity contracts can have additional diversification benefits when included in portfolios of traditional financial assets and then outline how portfolio optimization has been approached using metaheuristics.

Diversification Benefits of Commodity Futures

Diversification is best achieved through combining assets with low or even negative correlation into an investment portfolio. Due to the increased correlation among individual stocks, diversification possibilities of stock portfolios have become limited. Commodities in general and metals in

particular on the contrary have shown to yield low correlation with stocks (Jaffe, 1989; Chua et al., 1990; Hillier et al., 2006; Daskalaki and Skiadopoulos, 2011), especially because macroeconomic shocks tend to impact stock and commodity prices in different directions (Silvennoinen and Thorp, 2013). Particularly concerning inflation, the reaction of commodities and stocks might differ fundamentally. Indeed, while unexpected inflation leads to an increase in the prices of commodities, stocks have generally been found to be an inflation-protected asset class (Hardouvelis, 1987; McQueen and Roley, 1993) or, in case of fluctuation, have even yielded falling prices (Fama, 1981; Amihud, 1996; Bansal et al., 2014). However, investment in physical commodities is characterized by high costs (storage) and additional uncertainty (perishable nature, seasonal cycles of the goods) so that commodity futures are a natural alternative, providing the same diversification benefits without the implied disadvantages to an investor. Bansal et al. (2014) calculate the efficient frontier for an investment portfolio made up of indices of commodity futures and stocks and find it to lie above that for a traditional stock and bond portfolio, i.e., they show that for their data set any given return level a portfolio including commodity futures achieves a lower level of risk. This diversification takes place independent of the state of the stock market: Crude oil futures contracts were shown to lead to successful diversification in both upward and downward trending stock markets (Geman and Kharoubi, 2008). Commodities emerge as a significant diversifier of both equity returns and volatility (Brooks and Prokopczuk, 2013). Investment in commodities is also demonstrated to significantly improve investor's expected utility. In this regard, Garrett and Taylor (2001) find that expected-utility-maximizing investors, depending on the degree of risk aversion, should invest 30 to 68% of their wealth in commodities. Unlike in Geman and Kharoubi (2008), this finding is event-dependent and period-specific. In contrast to the above mentioned studies that were conducted from the standpoint of a US-based investor, Belousova and Dorfleitner (2012) show that a euro investor can also accrue diversification benefits from commodity investments. In particular, the authors emphasize that industrial metals, agricultural commodities and livestock contribute to the reduction of investment risk, while precious metals and energy are associated with both lower portfolio risk and higher return. Investments in commodities become especially rewarding when the general financial climate becomes negative (Chow et al., 1999). In the quest for hedging (a significant and negative response of metal returns to bond market returns irrespectively of the bond market stance) and 'save haven' vehicles against losses in the sovereign bond market, Agyei-Ampomah et al. (2014) highlight the superiority of industrial metals (aluminium, copper, lead, nickel, tin and zinc) over precious metals (gold, silver, platinum and palladium). Antonakakis and Kizys (2015) underline the information contents of gold, silver and platinum in improving forecast accuracy of returns and volatilities of palladium, crude oil and the EUR/CHF and GBP/USD exchange rates. Prominent among commodities is gold – commonly regarded as a 'safe haven' asset – that provides wealth protection by hedging investments in the stock and foreign exchange markets, even during extreme price movements during periods of turmoil (Pukthuanthong and Roll, 2011; Ciner et al., 2013; Reboredo, 2013). The above results have previously been confirmed by You and Daigler (2012). However, they employ individual stocks and futures contracts rather than stocks and commodity indices. This drastically increases the complexity of the optimization problem. In order to cope with this, they resort to a portfolio optimization software package, which is however limited to 120 observations. To circumvent this, a metaheuristic algorithm is applied in this paper that is not only capable of dealing with an extensive number of observations, but also with further constraints that are outlined in Section 3.

However, an extensive analysis on the diversification benefits of commodity futures by Cheung and Miu (2010) raises concerns about the universal validity of the above findings, indicating that individual assessments become necessary. Furthermore, the ex-post performance of stock and commodity futures portfolios was found to be inferior to that of a portfolio made up of traditional assets by Daskalaki and Skiadopoulos (2011), thus making this another important question in the evaluation. It is thus also evaluated whether the applied methodology is able to identify stable asset weights based on ex-post performance and if so, whether the performance of the portfolio including commodity futures outperforms the traditional stock portfolio.

Metaheuristics and Portfolio optimization

Whereas the original Markowitz formulation of the mean-variance optimization can be solved employing traditional optimization methods including, among others, linear programming (Mansini et al., 2014; Pae and Sabbaghi, 2014; Bruni et al., 2015), quadratic programming (Pachamanova

and Fabozzi, 2010; Sawik, 2012a) and stochastic programming (Pınar, 2007), metaheuristics have increasingly been used to solve richer variants of the original problem, including more realistic constraints and larger numbers of assets. Concerning optimality, Cesarone et al. (2013) were able to develop an exact increasing set algorithm that, for small instances, solves the POP with quantity and cardinality constraints optimally and can be extended into a heuristic procedure to account for larger instances. While metaheuristics do not guarantee finding a globally optimal solution, Gilli and Schumann (2012) point out that the goal real-life portfolio optimization tasks is not to provide an optimal solution, but one that fulfills the decision-maker's objectives to a highly satisfactory extent. There are notably several reasons for this development. Generally, traditional optimization approaches are prone to provide sub-optimal solutions when the expected returns or variances employed as inputs are characterized by estimation errors (Kolm et al., 2014). More importantly for this analysis is, however, the fact that traditional methods have been found to provide weak ex-post performance of portfolios (DeMiguel et al., 2009).

Metaheuristic methods for portfolio optimization have been designed to address the above theoretical and practical limitations of classical approaches. The following authors have significantly contributed to advancements in this field. Chang et al. (2000) consider a basic version without further constraints, introduce metaheuristics as a portfolio optimization solving methodology and highlight their potential use. The inclusion of constraints is brought forward by Derigs and Nickel (2003) and Derigs and Nickel (2004) who among other approaches developed a two-stage SA metaheuristic, in which they controlled for cardinality constraints and transaction costs through turnover volume restrictions. A more comprehensive model including sector capitalization constraints, minimum transaction lots and cardinality constraints is introduced by Soleimani et al. (2009). Their work recognizes the limitations that investors in practice pose further demands for their respective investments. While the traditional formulation of the model requires the asset weights to add up to one and prohibits short-selling, further constraints that have received overwhelming attention in the literature are cardinality, quantity and pre-assignment constraints. Cardinality constraints, first introduced by Schaerf (2002), ensure diversification and, as marginal diversification benefits decrease (Maringer and Kellerer, 2003; Maringer, 2005; Pachamanova and Fabozzi, 2010), limit managerial effort (Di Gaspero et al., 2011) and transaction costs in the form of bank and broker fees (Baule, 2010) by introducing a minimum and maximum threshold for the number of assets included in the portfolio. Quantity constraints confine the weights between a lower and upper bound to limit one-sided exposure, while ruling out negligibly small investments whose profits could be eroded by transaction costs. Lastly, pre-assignment constraints allow the investor to choose some assets to be included in the portfolio ex-ante regardless of their risk-return characteristics. An updated and detailed literature review on the application of metaheuristics to portfolio optimization is provided by Kolm et al. (2014).

3. Formal Description of the Problem

In this section, the mean-variance portfolio optimization model is described. First, the model parameters and the algorithm's in- and outputs and then a mathematical model are presented.

There is a set of potential assets to choose from. On the one hand, there is a set of n stocks $S = s_1, s_2, ..., s_n$ and on the other hand, we further include a set of m individual commodity futures contracts $F = f_1, f_2, ..., f_m$, resulting in a total number of potential assets $A = a_1, a_2, ..., a_{m+n}$ equal to $m + n$. For all assets, the expected return based on historical data of a specified time period $E[R_i]$ is calculated, so that $\forall i \in \{1, 2, ..., m + n\}$, a_i is assigned a known expected return $E[R_i]$ based on historical data. It is to be noted that, unlike previous research, we allow for the inclusion of such assets with negative expected returns for two reasons. The first is a technical one: As real-life investors choose from a potential pool of assets whose returns are notably influenced by the historical time horizon chosen for analysis, we prevent introducing a bias that the period of analysis might introduce. Furthermore, as we show in Section 6, the introduction of futures contracts with slightly negative returns can still cause the portfolio to outperform that composed of only stocks. As a measure of riskiness of the portfolio, its variance is calculated. In order to do so, risk indices between the individual pairs a_i, a_j are calculated: $\sigma_{ij} = \sigma_{ji} \forall R^*$. These individual risk indices consist of the covariances between the returns of pair of assets.

From the potential assets a portfolio is to be constructed. A portfolio can be interpreted as a vector of weights $W = w_1, w_2, ..., w_{m+n}$, in which each w_i represents the weight of the corresponding asset in the portfolio, i.e., the portion of the investor's budget invested into this asset; for

non-included assets the weight is equal to zero, so that $0 \leq w_i \leq 1$. As is common in the literature, it is further assumed that the full budget is to be invested, i.e., $\sum_{i=1}^{m+n} w_i = 1$.

The portfolio construction is then constrained by the following user-provided inputs.

- The expected return of the portfolio is bound to a minimum of $R > 0$.
- $\forall i \in \{1, 2, ..., m+n\}$, $z_i = 1$ if $w_i > 0$ and $z_i = 0$ otherwise. The cardinality constraint limits the total number of assets to be included in the portfolio $\sum_{i=}^{m+n} z_i$ between a lower value k_{min} and an upper value k_{max}. For the purpose of this study, the values are set to one and five, respectively.
- The quantity constraints limit the potential weights for each individual potential asset a_i between a lower value ϵ_i and an upper value δ_i that are set to 0.01 and 1 to prevent too small fractions of investment and thus increased managerial costs.
- Lastly, the investor can pre-select certain assets regardless of their risk-return characteristics based on personal preferences: $\forall i \in \{1, 2, ..., m+n\}$, $p_i = 1$ if a_i is pre-assigned into the portfolio and $p_i = 0$ otherwise.

More formally, the aforementioned constraints are to be respected while the optimal portfolio is chosen by determining the fractions of the assets so that the overall portfolio variance is minimized. The portfolio variance is a function of the variances of the individual assets as well as of the covariances of all pairs of returns on assets selected in the portfolio:

$$\sum_{i=1}^{m+n} \sum_{j=1}^{m+n} \sigma_{ij} w_i w_j, \tag{B.1}$$

subject to:

$$\sum_{i=1}^{m+n} r_i w_i \geq R^*, \tag{B.2}$$

$$\sum_{i=1}^{m+n} w_i = 1, \tag{B.3}$$

$$k_{min} \leq \sum_{i=1}^{m+n} z_i \leq k_{max}, \tag{B.4}$$

$$0 \leq \epsilon_i \leq \delta_i \leq 1, \forall i \in \{1, 2, ..., m+n\}, \tag{B.5}$$

$$z_i \in 0, 1, \forall i \in \{1, 2, ..., m+n\}, \tag{B.6}$$

$$z_i \leq M w_i, \forall i \in \{1, 2, ..., m+n\}, \tag{B.7}$$

$$p_i \leq z_i, \forall i \in \{1, 2, ..., m+n\}. \tag{B.8}$$

The last three equations concern the pre-assignment constraint. z_i is defined as a binary variable that depends on the parameter p_i. If the asset is pre-selected (i.e., $p_i = 1$), it must be included in the solution (i.e., $z_i = 1$) irrespective of its risk-return characteristics. Furthermore, M is a very large positive value such that $M w_i > 1$ for all i if $w_i > 0$.

4. Solving Methodology

We employ a solving matheuristic (Maniezzo et al., 2009) based on an iterated local search metaheuristic first applied by Martin et al. (1992) and the open-source quadratic programming solver ojAlgo (http://ojalgo.org), developed in Java, that, as a local search, determines the optimal weights for a given solution of assets. Furthermore, as to not call the solver repetitively for a given solution, a cache memory (implemented as a hash map data structure) is implemented that stores optimal weights for a given combination of assets. Fig. B.1 presents a high-level flowchart

of the approach. The algorithm was implemented as a Java application and all experiments were performed on a standard personal computer equipped with Intel Core i5 CPU at 3.2 GHz, 4GB RAM and Windows 8.

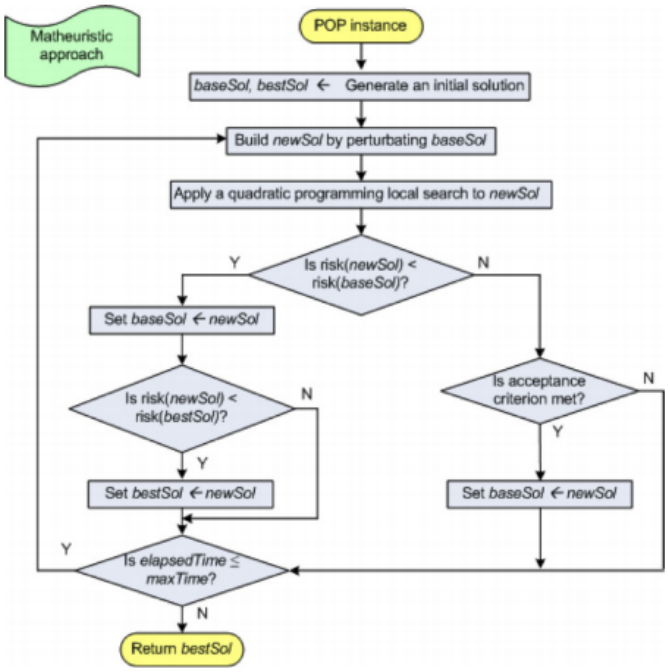


FIGURE B.1: Basic flowchart of the matheuristic

Initially, a first solution is generated by constructing a portfolio composed solely of the asset with the highest return, generally also resulting in considerable risk. If this solution does not achieve the minimum return, the instance is infeasible. This initial dummy solution is then assigned as base solution and best-found solution so far. The while-loop introduces the main iterative solution improvement procedure. It consists of three stages that successfully combine exploitation and exploration of the search space: a perturbation stage, a quadratic programming local search and an acceptance stage. The perturbation stage applies significant changes to the base solution through a shaking procedure that creates a new solution. This procedure consists of randomly erasing a number of non pre-selected assets in the solution and randomly introducing new assets until reaching k_{max}. The number of assets erased (k) is determined by the order of the current neighborhood. Each $newSol$ obtained trough the shaking procedure is provided to a solver to determine the optimal weights allocated to a given set of assets. If this has previously been done, the cached weights are called, if not the found optimal weights are cached correspondingly. Then, this new solution undergoes a local search (LS) phase in order to find the local minimum within the defined neighborhood structure. During the LS procedure, we randomly select the same k number of assets, and for each of those selected assets we order the remaining discarded ones according to their mutual risk characteristics with said asset. Then, randomly a non-preselected asset of the portfolio is replaced by a previously discarded one employing a biased randomization technique, which is relying on a geometric distribution with a parameter β that after pre-computational experiments has been set to $\beta = 0.3$. If the so-created new solution outperforms the base solution in terms of risk, it is accepted as new base solution. By the same token, this solution is then compared with the so-far-best solution that is updated based on the same risk criterion. However, under certain conditions (acceptance criterion) a worse solution might temporarily be accepted as base solution in order to further enhance exploration and escape local minima in the search space. We implemented a credit-based acceptance criterion that allows for degradation of the base solution if the previous improvement in the level of risk is not exceeded in absolute values. Finally, the best-found solution is returned. This provides the investment portfolio corresponding to a certain threshold return that provides the lowest-found corresponding level of risk.

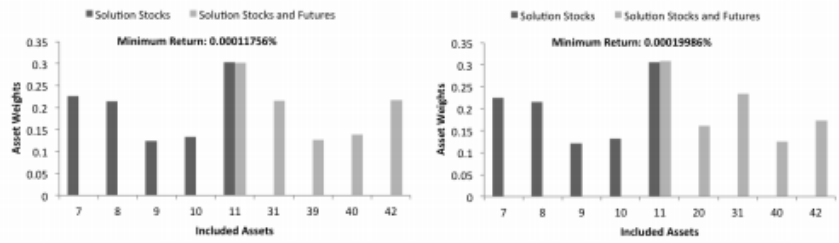


FIGURE B.2: Solutions for minimum returns on the lower spectrum

5. Data Description

As our approach is concerned with the comparison of a stocks-alone and a stocks-and-futures portfolio, we obtained individual daily historical closing price data for the Dow Jones 30 constituents on the one hand and daily settlement prices for the 21 most actively traded commodity futures prices in the United States as proposed by You and Daigler (2012) covering the period from February 18, 2014 to April 1, 2016, resulting in 535 observations for each time series. The sample period has been chosen due to the availability of the individual time series so that the data sample does not yield missing values. Due to expiration of fixed-maturity futures contracts, the continuous series are created by data providers by rolling over the futures contracts of different maturities. Table B.1 presents the average daily returns and the corresponding standard deviations for each of the included stocks and commodity futures contracts.

The average correlations within the two asset classes, as well the mean overall correlation, are presented in Table B.2. More detailed correlation calculations are available from the authors on request. Due to the askew distribution of correlations these have been calculated by transforming the individual correlations into Fisher-Z-values, taking the arithmetic mean and then retransforming. It becomes obvious that the correlation between the potential stocks is significantly higher than that within the class of commodity future contracts. Furthermore, the mean correlation between stocks and futures is the lowest overall for the data sample. This reinforces the assumption that a combined portfolio of stocks and futures can lead to superior diversification and a resulting lowering of the associated expected risk for a given portfolio return.

6. Analysis of Results

In the following the results for two experiments are analyzed first with respect to risk analysis of the ex-ante portfolios and then with respect to the ex-post performance, or stability, of the found solutions. Ex-ante portfolios are those portfolios with constituent assets and weights determined by the iterative local search based on the data gathered for the sample period. Ex-post portfolios refer to the application of the ex-ante portfolio asset weights to the data following the sample period at time $t + 1$. It thus refers to a hypothetical investment at time t into the best-found portfolios that is then evaluated one month later at time $t + 1$.

Fig. B.2 and B.3 present two exemplary solutions. It is to be noted that assets 1 through 30 represent stocks and assets 31 through 51 represent commodity futures contracts. For low-level minimum returns in Fig. B.2, the first stock portfolio contains portions of asset 7, 8, 9, 10, and 11 (all stocks), while the stocks and futures portfolio contains assets 11, 31, 39, 40, and 42 (one stock and four futures contracts). For high-level minimum returns, the solutions are much more similar with respect to the asset composition. The first exemplary stock solution in Fig. B.3 is composed of assets 14, 20, 21, and 30 (all stocks), while the stocks and futures portfolio contains asset 39 instead of asset 21. The exemplary solutions showcase that the solutions for higher minimum returns overlap further and are more similar in terms of selected assets constituents and weights than for low levels of return. This illustrates the finding that the allocation of commodity futures increases with increasing risk aversion of the investor, yielding that they represent a valuable alternative as diversification means especially for lower-risk portfolios.

TABLE B.1: Descriptive statistics

Assets	Average Return	Standard Deviation
Stocks	**0.000307%**	**0.012996**
Apple	0.076993%	0.015544
Microsoft	0.084877%	0.015512
Exxon Mobil Corporation	-0.014890%	0.013191
Johnson & Johnson	0.035448%	0.009978
General Electric Company	0.047681%	0.012239
JPMorgan Chase & Co.	0.015270%	0.014037
The Procter & Gamble Company	0.013599%	0.009085
Verizon Communications Inc.	0.032659%	0.009721
Wal-Mart Stores Inc.	-0.010853%	0.011372
Pfizer Inc.	-0.004829%	0.011537
The Coca-Cola Company	0.038688%	0.009100
Chevron Corporation	-0.021950%	0.015983
Visa Inc.	0.069322%	0.014216
The Home Depot, Inc.	0.110242%	0.012426
The Walt Disney Company	0.050015%	0.012805
Merck & Co. Inc.	0.002151%	0.012748
International Business Machines Corporation	-0.026583%	0.012757
Intel Corporation	0.062516%	0.015464
Cisco Systems, Inc.	0.054523%	0.013921
UnitedHealth Group Incorporated	0.116388%	0.014094
McDonald's Corp.	0.058294%	0.010550
3M Company	0.050225%	0.010810
NIKE, Inc.	0.103030%	0.014542
The Boeing Company	0.005434%	0.014169
United Technologies Corporation	-0.017746%	0.011471
The Goldman Sachs Group, Inc.	0.005036%	0.013801
American Express Company	-0.061105%	0.013448
E. I. du Pont de Nemours and Company	0.009985%	0.015360
Caterpillar Inc.	-0.030999%	0.015346
The Travelers Companies, Inc.	0.067270%	0.009718
Commodity Futures	**-0.000496%**	**0.000338**
Brentcrudeoil	-0.120960%	0.025457
Copper	-0.043944%	0.012876
Crudeoil	-0.125399%	0.025594
Cocoa	0.017888%	0.012117
Coffee	-0.055508%	0.023227
Corn	-0.031191%	0.014505
Cotton#2	-0.058913%	0.013930
Feedercattle	-0.032845%	0.010810
Gold	-0.004478%	0.009594
Heatingoil	-0.120557%	0.023106
KCWheat	-0.079898%	0.016955
Leanhog	-0.054406%	0.023260
Livecattle	-0.035734%	0.011952
Naturalgas	-0.116111%	0.022118
Orangejuice	-0.003081%	0.020640
Silver	-0.017541%	0.016391
Soybean	-0.047312%	0.015006
Soybeanmeal	-0.030699%	0.020249
Soybeanoil	-0.042044%	0.013330
Sugar#11	0.015679%	0.022557
Wheat	-0.054146%	0.017646

TABLE B.2: Average correlations between the asset classes

Within Stocks	Within Futures	Stocks and Futures	Overall
0.4385	0.0765	0.0075	0.1756

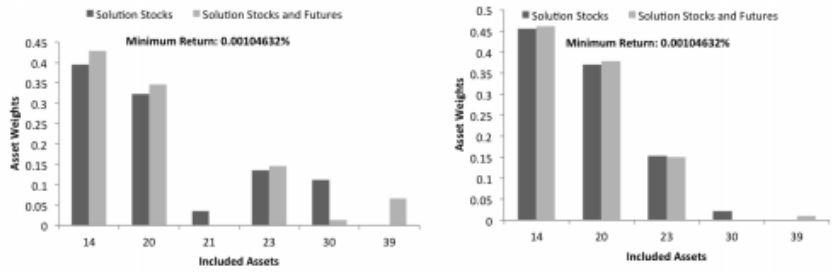


FIGURE B.3: Solutions for minimum returns on the higher spectrum

6.1 Risk Analysis

Table B.3 summarizes the results of the two experiments. It essentially compares the results obtained for a particular minimum return. At first sight, the previously mentioned complexity of the problem becomes obvious when the instance times are considered: They significantly increase for the composite portfolios that are selected from an asset pool of 51 potential constituents as opposed to the basic formulation that only considers a pool of 30 stocks. More importantly, the associated risk as expressed by the weighted covariances of the constituent assets' returns is presented. As expected, it continuously increases with increasing returns demanded by the investor. The risks between two best-found solutions based on different asset pools are then compared. The existence of a risk gap, or the difference in the risk objective function value for a same given minimum return between two solutions, is explained by the inclusion of individual futures contract in the best-found portfolio compared to the stock-alone portfolios. It is to be noted that this was always done at the expense of excluding at least one previously included stock and not through the addition of new assets in the portfolio, indicating no rise in managerial effort or transaction costs associated with including futures in a traditional stock portfolio. A positive risk gap indicates that the risk was minimized with respect to the solution found for a stock-alone portfolio and thus successfully diversified. This was the case for 94 out of the 99 return instances, while the remainder showed a gap equal to zero. This and the average percentage gap of 26.84% strongly reinforce the initial intuition that individual futures contracts increase diversification beyond that which can be achieved through stock diversification alone.

TABLE B.3: Results for a selected subset of minimum returns

	Stock-alone Portfolio		Stock-and-Futures Portfolio		Gap	Gap [%]
Minimum Return	**Risk (1)**	**Time [s]**	**Risk (2)**	**Time [s]**	**(1) - (2)**	**(1) - (2)**
0.0000117564	0.0000531088	0.873	0.0000218480	10.202	0.0000312608	58.86180821%
0.0000822945	0.0000531088	0.042	0.0000230232	0.567	0.0000300856	56.64899226%
0.0001528327	0.0000531088	0.031	0.0000238688	3.141	0.0000292400	55.05678908%
0.0002233709	0.0000533257	0.183	0.0000265600	0.279	0.0000267657	50.19287135%
0.0002939091	0.0000533474	0.058	0.0000271226	12.655	0.0000262247	49.15853444%
0.0003644473	0.0000540647	0.014	0.0000311877	2.541	0.0000228770	42.31411623%
0.0004349855	0.0000546027	0.287	0.0000343818	16.648	0.0000202209	37.03278409%
0.0005055236	0.0000558261	0.225	0.0000375692	8.41	0.0000182569	32.70316214%
0.0005760618	0.0000573681	0.099	0.0000428111	19.99	0.0000145571	25.37472916%
0.0006466000	0.0000601832	1.1	0.0000473472	11.772	0.0000128360	21.32821120%
0.0007171382	0.0000642364	0.254	0.0000543535	17.792	0.0000098829	15.38520216%
0.0007876764	0.0000694180	0.4	0.0000619521	6.255	0.0000074659	10.75499150%
0.0008582145	0.0000766848	3.21	0.0000705611	16.555	0.0000061238	7.98554603%
0.0009287527	0.0000855374	0.091	0.0000812718	13.253	0.0000042656	4.98682448%
0.0009992909	0.0000965467	1.369	0.0000936730	5.567	0.0000028737	2.97648703%
0.0010698291	0.0001099596	1.089	0.0001090107	1.553	0.0000009488	0.86295330%
0.0011403673	0.0001373443	0.001	0.0001373443	0.001	0.0000000000	0.00000000%

The five instances, in which there was no difference between the asset constituents of the portfolios, were the ones with the highest required minimum returns. Generally, the gap decreased with increasing minimum returns. This relationship is further demonstrated in Fig. B.4,in which the absolute risk gaps are depicted by the horizontal distance between the graphs corresponding to stock-alone and stock-and-futures portfolios respectively.

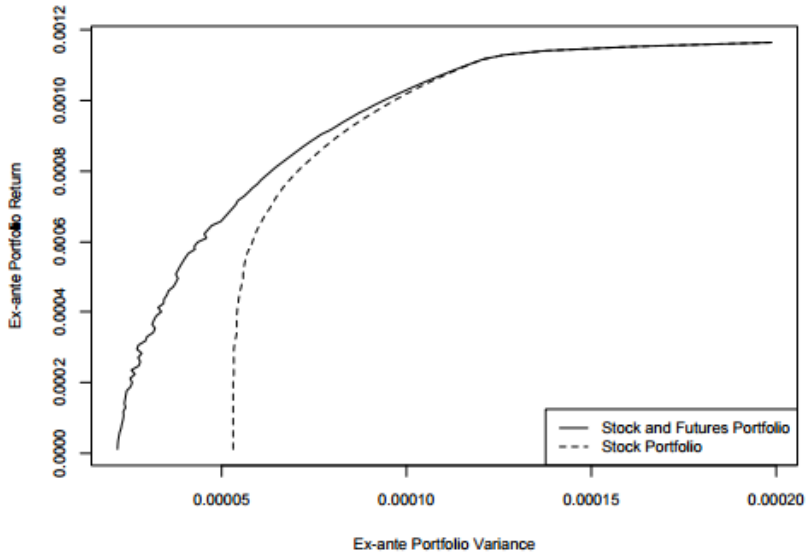


FIGURE B.4: Depiction of ex-ante minimum return and associated risk for the two asset pools

The graphs provide two conclusions. On the one hand, there is a threshold return, beyond which additional returns require a more significant increase in associated risk because this return is generally only achieved by fewer assets, reducing diversification benefits. For this set of data, it is found at 0.00114%. From this threshold on, the minimum required return could solely be achieved by certain assets, thus reducing the pool of potential assets and leading to portfolios of fewer included assets than the maximum number of five dictated by the cardinality constraint. This leads to portfolios composed of only stocks and thus also to a zero gap between the two portfolio types. On the other hand, it becomes obvious that the gap is much larger for low-return portfolios, from which one can conclude that risk-averse investors profit to a larger extent from futures diversification.

To better illustrate these conclusions Fig. B.5 presents three box plots of the percentage risk gaps. The lower one presents the distribution of the risk gaps for the overall sample of optimal portfolios. The circle represents the mean of 26.84%. The immense diversification potential is provided not only by the high maximum value of 58.9%, but also the high values associated with the different quantiles. We further divided the portfolios into low-return and high-return portfolios. The upper two box plots quantify the difference in diversification benefits between low-return and high-return requirements. It becomes obvious that diversification benefits are increased for risk-adverse investors and thus portfolios on the lower required minimum return spectrum.

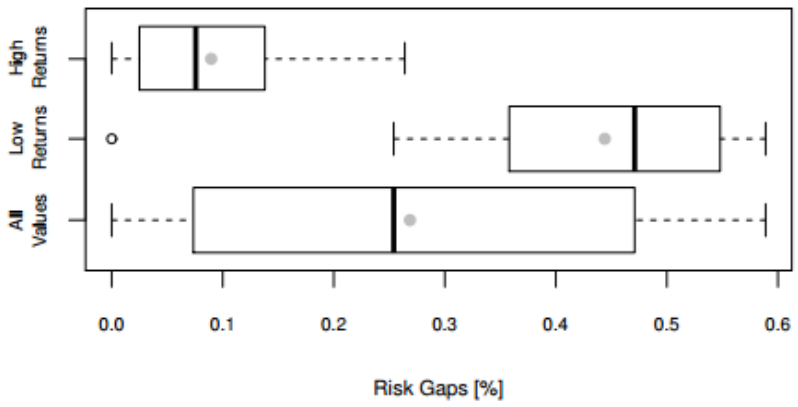


FIGURE B.5: Risk gaps between the best stocks portfolios and the best futures portfolios

We can thus conclude that, on the one hand, our algorithm is able to find portfolios that fulfill the investor's constraints in reasonable computing times of a few seconds and, on the other, the diversification benefits of individual futures contracts are immense. This has been shown by the risk level gaps between an stocks-alone portfolio and one that included futures, which were significant and never negative.

6.2 Ex-post Stability Analysis

Concerning the stability of the resulting portfolios, we have conducted an ex-post application of the resulting portfolio weights. Two of these are exemplarily presented below; the first corresponds to the lowest daily minimum return and the second corresponds to the highest minimum return level at which the portfolios still differed. We conducted a one-month ahead analysis and compared the resulting returns of the portfolio with the corresponding minimum return on a monthly basis. The ex-post analysis consists of the hypothetical investment into the best-found solution at the corresponding asset weights at the end of the sample period. Then, after a holding period of one month, the returns on the investment are calculated based on the actual price movements of the constituent assets.

TABLE B.4: Ex-post performance of two exemplary solutions

	Solution 1 – Low Return	Solution 2 – High Return
Required Daily Return	0.0000118%	0.0011051%
Actual Return Stocks Portfolio	-0.0009301%	0.0002369%
Actual Return Stocks and Futures Portfolio	0.0051194%	0.0003352%

Table B.4 presents the metrics of the ex-post analysis for the first of two best-found portfolios presented in Fig. B.2 and B.3 respectively. Solution 1 was found for an extremely risk-averse investor, while solution 2 represents a risk-loving investor's investment recommendation. The daily minimum return was a user-defined input for the proposed matheuristic. The actual return is presented below the minimum return for both the stocks and the combined stocks and futures portfolio. Exemplified by the two solutions in Table B.4 the ex-post performance differs greatly depending on the required minimum return and the asset pool.

Fig. B.6 presents the risk-return characteristics of all ex-post portfolios for the ex-post period. At first sight, it can be constated that, solely taking into consideration the positive portion of the return axis, the two curves resemble the shape of a Markowitz efficient frontier curve.It further becomes obvious that stock-and-futures portfolios overall achieved positive average ex-post daily returns, while a large portion of stock-alone portfolios presents the potential investor with negative returns. Because these negative returns are the result of downside risk exposure of the accompanying portfolios with high variance, it is intuitive that this part of the plot does not possess a positive slope. Concluding, it becomes evident that adding futures to the portfolios significantly improved the portfolio's behavior with respect to traditional financial theory in that increased returns can be achieved by assuming a more risk-exposed investment position. Moreover, the superiority of the stock and futures portfolios in ex-post performance already established in Fig. B.5 is reinforced when considering both investment dimensions, risk and returns. Fig. B.6 shows that the ex-post portfolios of stocks and commodity futures can be a more effective vehicle of diversification than the portfolios of stocks only. Indeed, the ex-post portfolio variance is smaller for the former than for the latter, as shown by the minimum variance portfolio. Moreover, for a given value of the portfolio variance, average returns are larger for the portfolio combining both stocks and commodity futures. Furthermore, including commodity futures caters not only to risk-averse but also to risk-taking investors, since a broader range of values for both portfolio variance and return can be obtained.

In the following, it is analyzed whether the remaining instability of the portfolio weights was caused primarily by the risk or the return dimension of the portfolios.

Fig. B.7 depicts the return dimension and presents a contrast of the minimum required return and the ex-post achieved one. If they are identical, the points should lie on the 45° line through the origin. However, due to the generally volatile nature of financial returns, this is not the case. The ex-post stock-alone portfolios significantly underperform the ex-ante portfolios and yield negative average daily returns for low and medium risk portfolios. Furthermore, while the ex-ante portfolio

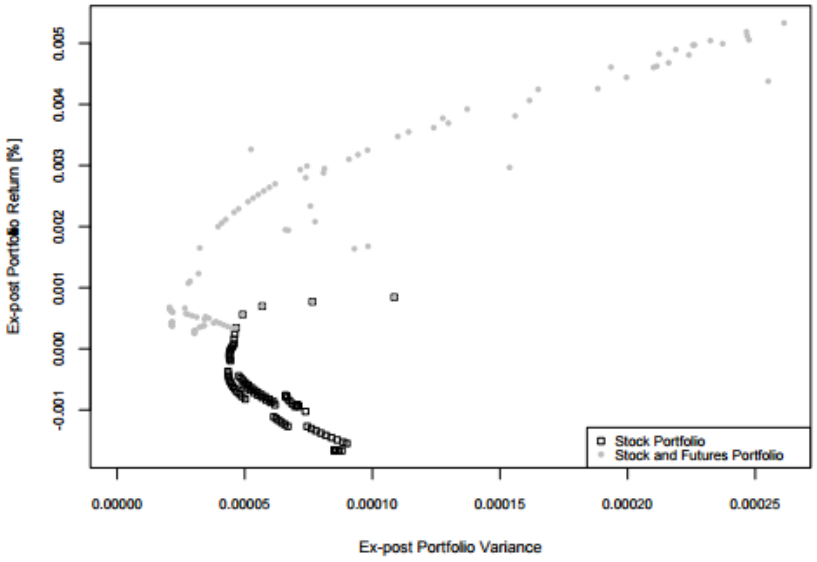


FIGURE B.6: Frontiers of ex-ante optimal portfolios in ex-post analysis

weights provide stable portfolio returns in that the minimum return is outperformed by the stock-and-futures portfolios for minimum returns on the lower spectrum of the analysis, the best-found solutions do not provide the minimum returns for extraordinarily high returns. While the latter result is a drawback to the investment, it is somewhat intuitive, as returns of such dimensions can only be achieved by assuming a significant level of risk. Moreover, the differences in the asset weights become almost negligible for minimum returns on the high end of the analysed spectrum as graphically shown in Fig. B.4. More strikingly, however, are the results for low return levels and thus risk-averse investors. Not only did the combined stock-and-futures portfolios generally outperform the traditional stock portfolio in the ex-post analysis, but also were they generally the only ones achieving an ex-post return of at least the originally required minimum return. This therefore reinforces the importance of futures diversification not solely from a risk perspective, but also from the perspective of stable short-term returns. However, the benefits of the portfolio optimization appear to be limited to lower risk portfolios and thus risk-averse investors from a return dimension perspective.

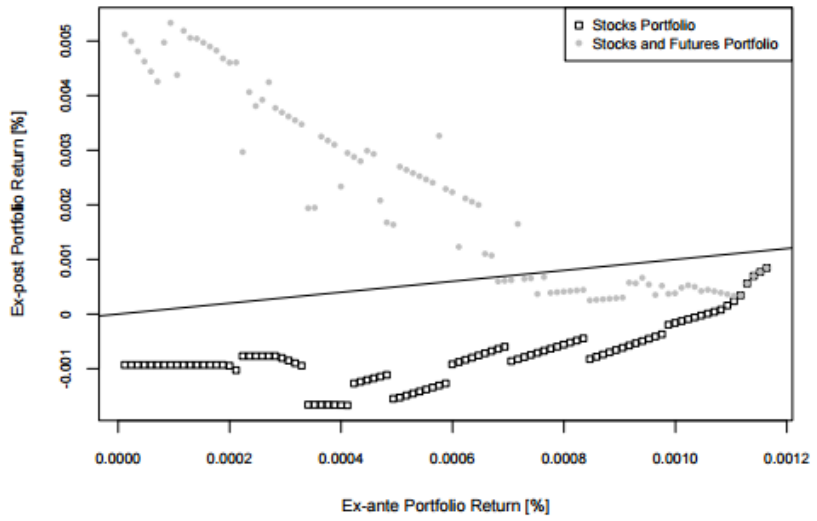


FIGURE B.7: Comparison of ex-ante minimum required returns and ex-post actual returns

Since the return dimension yields relatively stable performance for stock-and-futures portfolios, while this is not the case for stock-alone portfolios, it can be constated that this is a significant factor in distinguishing the different ex-post performances. Because, however, the ex-post combined portfolios also underperformed their ex-ante counterparts, the risk dimension is analyzed next. Overall, the non-stable risk variable of the portfolios seems to outweigh the volatility in returns in causing ex-post portfolio instability. Fig. B.8 presents a comparison of ex-ante risk and ex-post risk of the respective portfolios. If they were identical, the plotted points would lie on the 45° graph through the origin. As becomes obvious, low-levels of ex-ante risk are characterized by relatively non-stable ex-post risk levels and high variability. Contrary to that, high ex-ante risk level portfolios generally translate into lower-risk ex-post portfolios. It is to be noted further that the variability of the risk of stock-and-futures portfolios is higher than that of the stock portfolios. This variability then explains the level of ex-post instability of the combined portfolios.

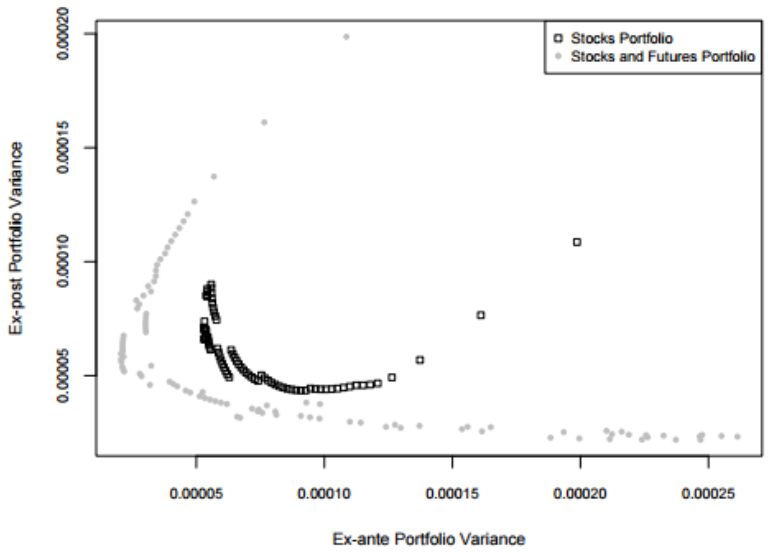


FIGURE B.8: Comparison of ex-ante and ex-post portfolio risks

Concluding, it can be stated that the best-found portfolios from Section 6.1 provide investors with stability in that the stock-and-futures portfolios outperform stock-alone portfolios in that they provide lower risk for a given minimum return level. However, the risk and return characteristics of the individual portfolios have shown to be instable over the observed period. While the return dimension mainly causes the difference in performance between the ex-post stock-alone and stock-and-futures portfolios, the risk dimension explains the remaining instability between the ex-ante and ex-post performance of the combined portfolios.

7. Conclusions and Future Works

This work has addressed a rich and original variant of the portfolio optimization problem (POP), in which both the individual constituents of the Dow 30 index as well as individual commodity futures contracts are considered as potential portfolio constituents. A short review of the literature has shown that the inclusion of futures is promising in reducing the risk level of a stock portfolio. Likewise, we have provided examples that highlight the use of matheuristic approaches in constructing promising portfolios. Then, a mathematical formulation for the rich POP has been described. In order to solve it, a matheuristic combining an iterated local search framework and a quadratic solver has been proposed. To the authors' knowledge this type of methodology has not previously been employed to analyze the diversification benefits of stock and futures portfolios. The computational experiments that were performed led to the conclusion that the solutions change in terms of expected risk when varying the pool of potential assets to include futures. For investors, this result yields the conclusion that futures contracts provide successful investment diversification. Particularly, risk-averse investors can drastically reduce their expected risk exposure by diversifying into stock-and-futures portfolios. Likewise, these portfolios of risk-averse investors yield more stable actual returns in the short term. The ex-post analysis has provided the

following conclusions of particular interest for portfolio managers: There is a difference in performance between the ex-post stock-alone and stock-and-futures portfolios mainly caused by the return dimension. Further, the risk dimension explains the remaining instability between the ex-ante and ex-post performance of the stock-and-futures combined portfolios. Thus,it is of interest to develop applications that provide more stable portfolio weights.

Some further future research lines remain open: *(i)* to extend the computational experiments so they include more instances and a deeper statistical analysis; *(ii)* to study the robustness of the solutions in front of small variations of the inputs in a sensitivity analysis; *(iii)* to include uncertainty in the optimization model to further account for the characteristics of financial markets and solve it with a simheuristic algorithm (Juan et al., 2015a); and *(iv)* to analyze the impact of the width of the sample period and associated bear and bull market activity periods.

Acknowledgements

This work has been partially supported with doctoral grants from the Open University of Catalonia, the Spanish Ministry of Economy and Competitiveness and FEDER (TRA2013-48180-C3-3-P, TRA2015-71883-REDT). Likewise we want to acknowledge the support received by the Department of Universities, Research & Information Society of the Catalan Government (Grant 2014-CTP-00001).

References

Agyei-Ampomah, S., D. Gounopoulos, and K. Mazouz (2014). "Does gold offer a better protection against losses in sovereign debt bonds than other metals?" In: *Journal of Banking and Finance* 40.1, pp. 507–521.

Amihud, Y. (1996). "Unexpected inflation and stock returns revisited–evidence from Israel". In: *Journal of Money, Credit and Banking* 28.1, pp. 22–33.

Antonakakis, N. and R. Kizys (2015). *Dynamic spillovers between commodity and currency markets*.

Bansal, Y., S. Kumar, and P. Verma (2014). "Commodity futures in portfolio diversification: Impact on investor's utility". In: *Global Business & Management Research* 6.2, pp. 112–121.

Baule, R. (2010). "Optimal portfolio selection for the small investor considering risk and transaction costs". In: *OR Spectrum* 32.1, pp. 61–76.

Beasley, J. E. (2013). "Portfolio optimisation: Models and solution approaches". In: *Tutorials in operations research* 10, pp. 201–221.

Belousova, J. and G. Dorfleitner (2012). "On the diversification benefits of commodities from the perspective of euro investors". In: *Journal of Banking and Finance* 36.9, pp. 2455–2472.

Brooks, C. and M. Prokopczuk (2013). "The dynamics of commodity prices". In: *Quantitative Finance* 13.4, pp. 527–542.

Bruni, R., F. Cesarone, A. Scozzari, and F. Tardella (2015). "A linear risk-return model for enhanced indexation in portfolio optimization". In: *OR Spectrum* 37.3, pp. 735–759.

Cesarone, F., A. Scozzari, and F. Tardella (2013). "A new method for mean-variance portfolio optimization with cardinality constraints". In: *Annals of Operations Research* 205.1, pp. 213–234.

Chang, T.-J., N. Meade, J. E. Beasley, and Y. M. Sharaiha (2000). "Heuristics for cardinality constrained portfolio optimisation". In: *Computers & Operations Research* 27.13, pp. 1271–1302.

Cheung, C. S. and P. Miu (2010). "Diversification benefits of commodity futures". In: *Journal of International Financial Markets, Institutions and Money* 20.5, pp. 451–474.

Chong, J. and J. Miffre (2010). "Conditional correlation and volatility in commodity futures and traditional asset markets". In: *Journal of Alternative Investments* 12.3, pp. 61–75.

Chow, G., E. Jacquier, M. Kritzman, and K. Lowry (1999). "Optimal portfolios in good times and bad". In: *Financial Analysts Journal* 55.3, pp. 65–73.

Chua, J. H., G. Sick, and R. S. Woodward (1990). "Diversifying with gold stocks". In: *Financial Analysts Journal* 46.4, pp. 76–79.

Ciner, C., C. Gurdgiev, and B. M. Lucey (2013). "Hedges and safe havens: An examination of stocks, bonds, gold, oil and exchange rates". In: *International Review of Financial Analysis* 29, pp. 202–211.

Cura, T. (2009). "Particle swarm optimization approach to portfolio optimization". In: *Nonlinear Analysis: Real World Applications* 10.4, pp. 2396–2406.

Daskalaki, C. and G. Skiadopoulos (2011). "Should investors include commodities in their portfolios after all? New evidence". In: *Journal of Banking and Finance* 35.10, pp. 2606–2626.

DeMiguel, V., L. Garlappi, and R. Uppal (2009). "Optimal versus naïve diversification: How inefficient is the 1/N portfolio strategy?" In: *Review of Financial Studies* 22.5, pp. 1915–1953.

Derigs, U. and N. Nickel (2003). "Meta-heuristic based decision support for portfolio optimization with a case study on tracking error minimization in passive portfolio management". In: *OR Spectrum* 25.3, pp. 345–378.

Derigs, U. and N. H. Nickel (2004). "On a local-search heuristic for a class of tracking error minimization problems in portfolio management". In: *Annals of Operations Research* 131.1-4, pp. 45–77.

DeSarbo, W. S. and W. L. Cron (1988). "A maximum likelihood methodology for clusterwise linear regression". In: *Journal of classification* 5.2, pp. 249–282.

Fama, E. F. (1981). "Stock returns, real activity, inflation, and money". In: *The American Economic Review* 71.4, pp. 545–565.

Garrett, I. and N. Taylor (2001). "Portfolio diversification and excess comovement in commodity prices". In: *The Manchester School* 69.4, pp. 351–368.

Geman, H. and C. Kharoubi (2008). "WTI crude oil futures in portfolio diversification: the time-to-maturity effect". In: *Journal of Banking and Finance* 32.12, pp. 2553–2559.

Gilli, M. and E. Schumann (2012). "Heuristic optimisation in financial modelling". In: *Annals of operations research* 193.1, pp. 129–158.

Golmakani, H. R. and M. Fazel (2011). "Constrained portfolio selection using particle swarm optimization". In: *Expert Systems with Applications* 38.7, pp. 8327–8335.

Hardouvelis, G. A. (1987). "Macroeconomic information and stock prices". In: *Journal of Economics and Business* 39.2, pp. 131–140.

Hillier, D., P. Draper, and R. Faff (2006). *Do precious metals shine? An investment perspective*.

Jaffe, J. F. (1989). "Gold and gold stocks as investments for institutional portfolios". In: *Financial Analysts Journal* 45.2, pp. 53–59.

Jensen, G. R., R. R. Johnson, and J. M. Mercer (2002). "Tactical asset allocation and commodity futures". In: *The Journal of Portfolio Management* 28.4, pp. 100–111.

Juan, A. A., J. Faulin, S. Grasman, M. Rabe, and G. Figueira (2015a). "A review of simheuristics: extending metaheuristics to deal with stochastic optimization problems". In: *Operations Research Perspectives* 2, pp. 62–72.

Kolm, P. N., R. Tütüncü, and F. J. Fabozzi (2014). "60 years of portfolio optimization: Practical challenges and current trends". In: *European Journal of Operational Research* 234.2, pp. 356–371.

Maniezzo, V., T. Stützle, and S. Vo (2009). *Matheuristics: Hybridizing metaheuristics and mathematical programming*. 1st. Springer Publishing Company, Incorporated.

Mansini, R., W. Ogryczak, and M. G. Speranza (2014). "Twenty years of linear programming based portfolio optimization". In: *European Journal of Operational Research* 234.2, pp. 518–535.

Maringer, D. (2005). *Portfolio management with heuristic optimization*. Berlin: Springer.

Maringer, D. and H. Kellerer (2003). "Optimization of cardinality constrained portfolios with a hybrid local search algorithm". In: *OR Spectrum* 25.4, pp. 481–495.

Markowitz, H. (1952). "Portfolio selection". In: *The journal of finance* 7.1, pp. 77–91.

Martin, O., S. W. Otto, and E. W. Felten (1992). "Large-step Markov chains for the TSP incorporating local search heuristics". In: *Oper. Res. Lett.* 11.4, pp. 219–224.

McQueen, G. and V. V. Roley (1993). "Stock prices, news, and business conditions". In: *Review of Financial Studies* 6.3, pp. 683–707.

Pachamanova, D. A. and F. J. Fabozzi (2010). *Simulation and Optimization in Finance*. John Wiley and Sons.

Pae, Y. and N. Sabbaghi (2014). "Log-robust portfolio management after transaction costs". In: *OR Spectrum* 36.1, pp. 95–112.

Patrick, H. T. (1966). "Financial development and economic growth in underdeveloped countries". In: *Economic Development and Cultural Change* 14.2, pp. 174–189.

Pınar, M. (2007). "Robust scenario optimization based on downside-risk measure for multi-period portfolio selection". In: *OR Spectrum* 29.2, pp. 295–309.

Pukthuanthong, K. and R. Roll (2011). "Gold and the Dollar (and the Euro, Pound, and Yen)". In: *Journal of Banking and Finance* 35.8, pp. 2070–2083.

Reboredo, J. C. (2013). "Is gold a safe haven or a hedge for the US dollar? Implications for risk management". In: *Journal of Banking & Finance* 37.8, pp. 2665–2676.

Sawik, B. (2012b). "Downside risk approach for multi-objective portfolio optimization". In: *Operations Research Proceedings*. Ed. by D. Klatte, H.J. Luthi, and K. Schmedders. Springer, pp. 191–196.

Schaerf, A. (2002). "Local search techniques for constrained portfolio selection problems". In: *Computational Economics* 20.3, pp. 177–190.

Silvennoinen, A. and S. Thorp (2013). "Financialization, crisis and commodity correlation dynamics". In: *Journal of International Financial Markets, Institutions and Money* 24, pp. 42–65.

Soleimani, H., H. R. Golmakani, and M. H. Salimi (2009). "Markowitz-based portfolio selection with minimum transaction lots, cardinality constraints and regarding sector capitalization using genetic algorithm". In: *Expert Systems with Applications* 36.3, pp. 5058–5063.

Talbi, E.-G. (2009). *Metaheuristics: From design to implementation*. New Jersey: John Wiley & Sons.

Woodside-Oriakhi, M., C. Lucas, and J. E. Beasley (2011). "Heuristic algorithms for the cardinality constrained efficient frontier". In: *European Journal of Operational Research* 213.3, pp. 538–550.

You, L. and R. T. Daigler (2012). "A Markowitz optimization of commodity futures portfolios". In: *The Journal of Futures Markets* 33.4, pp. 343–368.

B.6 Combining simulation with metaheuristics in distributed scheduling problems with stochastic processing times

Sara Hatami[1], Laura Calvet[1], Víctor Fernandez-Viagas[2], Jose Framinan[2], Angel A. Juan[1]

1. Computer Science Department, Open University of Catalonia-IN3, Ave. Carl Friedrich Gauss, 08860, Castelldefels, Spain
e-mail: {shatami,lcalvetl,ajuanp}@uoc.edu
2. Industrial Management, School of Engineering, University of Seville, Ave. Descubrimientos, E41092 Seville, Spain
e-mail: {vfernandezviagas,framinan}@us.es;

Abstract

This paper addresses a distributed scheduling problem with stochastic processing times where a product composed of several intermediate products has to be assembled at a particular moment. The intermediate products are processed in independent distributed manufacturing factories, and each factory can be modelled as a permutation flowshop. In our case, the processing times of the stages in the flowshops are random variables following a given probability distribution. The objective is to find robust job sequences for the factories. Three simheuristic algorithms are proposed, which consider the minimization of a specific measure: the makespan, the expected makespan or the makespan percentile. A set of computational experiments are carried out to illustrate this realistic and rich problem and the proposed methodology, and to compare their outputs under different levels of stochasticity.

Keywords: distributed scheduling problem, stochastic scheduling problems, simheuristics, metaheuristics, combinatorial optimization, horizontal cooperation.

1. Introduction

Nowadays, the manufacturing industry faces important challenges at the global level, including fierce competitiveness, short product life cycles, increasing speed of product innovation, high product variety and quality, and rising customer expectations, among others. Industries need to find proper strategies to cope with these challenges and remain successful in the market, being one of these strategies the use of distributed manufacturing systems (Moon et al., 2002), with contrasted benefits in terms of higher product quality, lower production costs and fewer management risks (Wang, 1997; Chan et al., 2005; Kahn et al., 2013).

In distributed manufacturing systems there is an horizontal cooperation among entities when they have strategic relationships and join their individual strengths to achieve a common goal, so the complexity of manufacture is shared among different entities, resulting in conditions in which risks and costs become acceptable and market opportunities can be captured, as quite often single manufacturing centers are not able to produce products within reasonable costs and increase product diversity because of rigid organizational structures, deterministic approaches to take decisions, lack of technology and a competencies' hierarchical allocation (Sluga et al., 1998; Wang et al., 2006). As a result, nowadays single manufacturing centers are infrequent while distributed manufacturing systems are quite usual (Moon et al., 2002; Naderi and Ruiz, 2010).

On the one side, constructing these collaborative manufacturing systems help industries to face market global challenges in an efficient way. On the other side, the optimization of these systems is more complicated. However, this optimization usually has a significant effect on production performance. The production operations (including scheduling in each entity) of these systems are organized in such a way that the needs of the entire system are covered and its objectives accomplished. The optimization of these systems has received a considerable attention from practitioners and the research community in recent years.

A well-established problem in this context is the so-called Distributed Permutation Flowshop Scheduling Problem (DPFSP) (Naderi and Ruiz, 2010). The DPFSP consists of a set of distributed manufacturing factories with flowshop configurations. The responsibility of the factories is to produce a product composed of various jobs. Each factory has to process a certain number of jobs, and all of them should be completed at a given deadline or before. Typically, the DPFSP involves

two decisions: assigning each job to be manufactured to a factory, and determining a job sequence for each factory.

Furthermore, the classical DPFSP assumes a static environment and deterministic processing times to simplify the problem. However, real-world manufacturing systems are dynamic and often exposed to uncertainties and unforeseen events such as machine breakdown, changing due date, operator unavailability, materials out of stock, order rush, etc (Rodammer and White, 1988).

Our paper addresses a problem related to the DPFSP. We assume that the components have already assigned, and deal with job sequencing with each flowshop. Furthermore, the processing times of the components in each one of the flowshops is a random variable. The objective is to find a robust job sequence for each factory which starts to process at the latest possible time while completes all jobs respected to the deadline. Since stochastic processing times are considered, we will only be able to guarantee that jobs are finished by then with a given probability. This probability depend on the probability of each factory ending on time. Thus, if a minimum probability is required, each PFSP can not be separately solved.

The problem under consideration can be also related to the Permutation Flowshop Scheduling Problem with Stochastic Times (PFSPST). The literature on this problem is not extensive, especially when compared with the PFSP (Lin et al., 2015; Fernandez-Viagas and Framinan, 2015b; Fernandez-Viagas and Framinan, 2015c; Hsu et al., 2015), this problem is becoming more popular (Baker and Altheimer, 2012; Kianfar et al., 2012; Juan et al., 2014a). Since the PFSP is an $\mathcal{NP}$-Hard problem when the number of machines are equal or higher than 3 (Garey et al., 1976), it is clear that our problem is also $\mathcal{NP}$-Hard. As a consequence, it is sensible to focus on designing heuristic or metaheuristic approaches for obtaining good solutions in reasonable CPU times.

More specifically, we adopt an approach that relies on a simheuristic algorithm for solving the PFSPs. Simheuristics are a class of efficient optimization algorithms that extend metaheuristics in a natural and flexible way by integrating simulation (Juan et al., 2015a). While there are some methodologies for solving the PFSPST, most present some shortcomings such as making hard assumptions on probability distributions of processing times or being able to solve only small-sized instances. In contrast, simheuristics constitute a simple approach able to cope with these shortcomings.

The rest of the paper is organized as follows: the next section defines the problem. A literature review is provided in Section 2. Section 3 describes our solving approach. Section 4 presents the computational experiments carried out, whereas the results are in Section 5. Finally, Section 6 offers some conclusions.

2. Problem definition

In our problem, there is a set F of f distributed manufacturing factories (Figure B.1). The shop configuration of each factory is a Permutation Flowshop Scheduling Problem (PFSP), which is a particular case of the Flowshop Scheduling Problem (FSP) (Johnson, 1954). In the FSP, there is a set M of m machines where each job of a set N of n jobs must be processed on each machine. Each job starts to process from the first machine to the last one. Therefore, the number of operations per job is equal to the number of machines. The j^{th} operation of job i is processed on machine j, and can start if the $j-1^{th}$ operation on machine $j-1$ has been completed and machine j is free. Processing times are supposed to be known in advance and deterministic. Other classical assumptions (Baker, 1974) are: *(i)* all operations and jobs are independent and available for processing at time 0; *(ii)* all machines are continuously available and there are no breakdowns; *(iii)* each machine can process at most one job at a time; *(iv)* each job can be processed in only one machine at a time; *(v)* once an operation of a given job on a given machine has started, it cannot be interrupted (i.e., no preemption is allowed until the processing has been completed); *(vi)* setup and removal times are sequence-independent and are included in the processing times or are negligible; and *(vii)* in-process storage is considered infinite. In the FSP, there are $(n!)^m$ possible solutions since the number of job permutations per machine is $n!$. There is a simpler version of this problem called Permutation FSP (PFSP) which assumes that all machines have the same job permutation and the job passing is not allowed. This problem has $n!$ possible solutions.

In this manufacturing layout, a product consisting of various components (jobs) has to be processed on the machines located at the factories. The processing time of each job i in each machine j, P_{ij}, is considered a random variable following a non negative probability distribution,

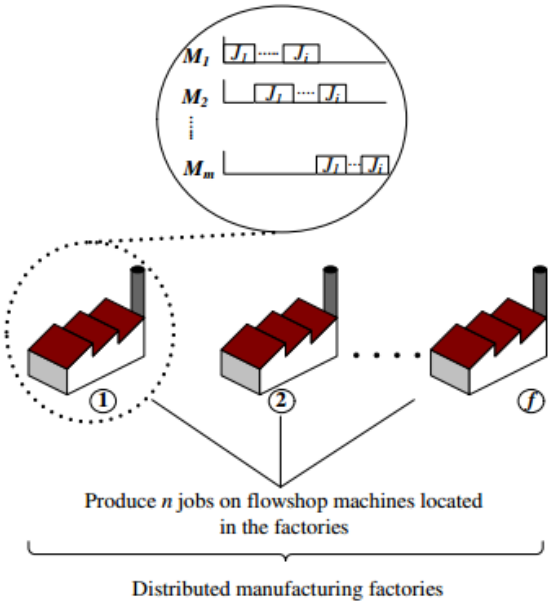


FIGURE B.1: A schematic diagram of the DPFSPST.

either theoretical or empirical. The product is considered finished when all its jobs have been completed.

It is required that all components of the product are completed by a (deterministic) deadline $\tilde{d}$ with a probability not lesser than p. Note that this common deadline represents a realistic consideration when the jobs should be later assembled in a final stage either in the same company or by a customer.

Consequently, it is intended that the processing operations for job i at factory k should terminate by the deadline $\tilde{d}$. In a PFSP with a deadline, a specific job sequence has a makespan associated and the starting time can be set at the deadline minus the makespan. In contrast, the PFSPST is characterized by having potential different makespans under different conditions for a given job sequence. Therefore, in our setting, at least one of the three following approaches could be considered:

- To ignore the stochastic nature of the problem and replace the random variables by their representation (typically their mean). In our case, these means solving a deterministic version of the problem using the mean processing times of the jobs and minimize the makespan. While ignoring the stochasticity may provide solutions of poor quality, it is not necessarily the case (see e.g. Framinan and Perez-Gonzalez, 2015). This is due to the fact that a deterministic optimization algorithm is faster and, as a consequence, may visit more solutions during a limited amount of time. Thus, if the level of stochasticity is low, there is a chance that solutions found are robust enough to have a good performance in a stochastic environment. This approach is labelled as *makespan* (M) in the following.

- To minimise the expected makespan. This approach stresses the average behaviour of the layout. However, if the starting time is set at the deadline minus the expected makespan, there is no guarantee that all processing operations will be completed on time. This approach is labelled as *expected makespan* (EM) in the following.

- To ensure that the final product will be finished on time with a probability p. This option allows the decision-maker to include a restriction that set the probability of finishing on time or, conversely, the risk of a delay. This approach is labelled as *percentile makespan* (PM) in the following.

  Since we assume that the factories are independent among them, p can be computed as: $p = \prod_{k=1}^{f} p_k$, and by assuming an equal allocation of probabilities we have $p_k = \sqrt[f]{p}$. Therefore, the problem is equivalent to ensure that factory k will finish its jobs with a probability p_k. In order to do so, the p_k-th makespan percentile can be computed for each factory k given

a sample of makespans, and its starting time can be set to the deadline minus the makespan percentile.

Figure B.2 shows the concepts of starting time, expected makespan and makespan percentile. Clearly, the choice between the last two approaches depends on the risk-aversion of the decision-maker. For example, if the decision-maker prefers to focus on the worst outputs (i.e., the largest makespans), it is better to minimize the makespan percentile requiring a high probability. On the other hand, if she/he prefers to analyse the average case, she/he should focus on minimizing the expected makespan. In Section 4, we will present different variants of an algorithm for each approach.

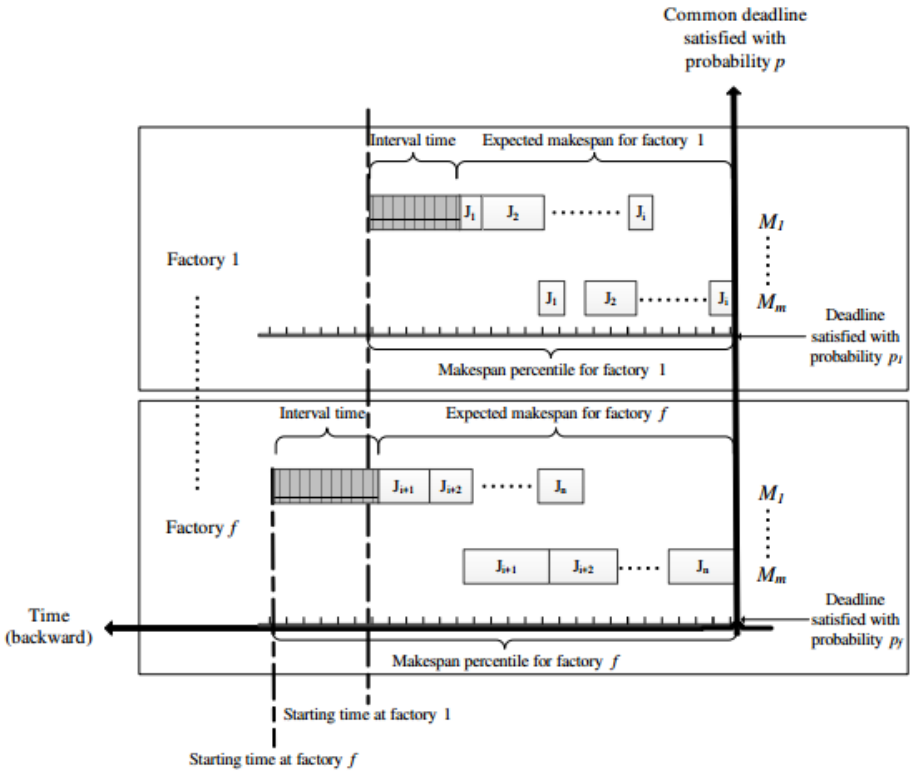


FIGURE B.2: Starting time, expected makespan and makespan percentile in the DPFSPST.

3. Literature review

In this section, we review the papers most related to the problem under study. To the best of our knowledge the problem has not been previously addressed. However, different streams of the literature are related to our problem, namely PFSPST, Distributed Flowshop Scheduling and Assembly Scheduling, the last ones assuming deterministic processing times. These are discussed in the next subsections.

Permutation Flowshop Problem with Stochastic Processing Times

While the PFSP has been intensively studied during the last few decades, the PFSPST has received less attention. Baker and Trietsch (2011) designed heuristics for addressing the 2-machine PFSPST, where the processing times are independent random variables following specific probability distributions. Later, Baker and Altheimer (2012) presented a methodology for the m-machine version. In addition, several variations of the PFSPST have been analyzed. For instance, Allaoui et al. (2006) and Choi and Wang (2012) worked on the stochastic hybrid flow shop scheduling problem, aiming to minimize the expected makespan. The same problem was tackled by Kianfar et al. (2012) with the goal of minimizing the average tardiness of jobs. A novel approach is applied

in Zhou and Cui (2008) for tackling the multi-objective stochastic PFSP, where both the flow time and delay time of jobs are minimized.

An interesting line is related to uncertainty. Basically, there are two categories: proactive (or robust) scheduling and reactive scheduling. For works falling in the first category, Roy (2010) propose constructing an original predictive schedule. The basic aim is to find schedules that do not require new schedules (or significant changes) when confronting disruptions. These works may consider probability distributions or sets of scenarios. Al Kattan and Maragoud (2008), Ghezail et al. (2010) and Liu et al. (2011) addressed the PFSP with uncertainty implementing proactive scheduling strategies. On the other hand, reactive scheduling consists in revising and re-optimizing schedules when unexpected events take place. A classical option is to obtain a predictive scheduling and then try to repair it according to the actual state of the system. A comprehensive review on rescheduling under disruptions is provided by Katragjini et al. (2013).

Some authors employ exact methods for addressing the PFSPST. A disadvantage of many of these methods is that they only work with a specific set of probability distributions and relatively small instances. Moreover, it may be difficult to adapt them for handling dependencies among processing times. Simulation techniques enable researchers to deal with these situations in a natural way. An interesting example is the work of Baker and Altheimer (2012), which proposed a hybrid approach combining heuristics and simulation. The authors tested three heuristic methods: two relying on the CDS heuristic (Campbell et al., 1970) and one on the NEH heuristic (Nawaz et al., 1983).

Distributed Flowshop Scheduling Problem

This problem has several similarities with the problem under study: there are f identical permutation flowshops where n jobs have to be processed. However, in this problem the jobs have not been assigned to each flowshop, so this assignment becomes part of the decision problem. Nowadays this problem is known as the Distributed Flowshop Scheduling Problem (DFSP) since Naderi and Ruiz (2010) resumed the topic for a distributed environment and makespan minimization. Nevertheless, this decision scheduling problem was first studied by David et al. (1996) based on a glass industry considering non-delay flowshops and batch production mode. Note that each factory is treated as line in this paper, but the mathematical scheduling problem inside is the same. Since then, it has been also studied under different names in the literature: *Parallel Flowline* (see e.g. Vairaktarakis and Elhafsi, 2000) and *Parallel Flowshops* (see e.g. Cao and Chen, 2003). Before Naderi and Ruiz (2010), the particular two-machine-flowshop layout in each factory or line has been solved using approximate algorithms by Zhang and Van De Velde (2012) and Al-Salem (2004). This particular problem turns to be a pure assignment problem due to the Johnson's rule (Johnson, 1954). For a general configuration of m machines, Naderi and Ruiz (2010) have proposed and compared several mixed integer linear programming models and constructive heuristics to solve the problem. Regarding iterated optimisation algorithm, the problem has received an increasing attention for makespan minimization in the literature in the last years. Gao and Chen (2011) have proposed a Genetic Algorithm using local search phases based on interchange and insertion of jobs. A Tabu Search algorithm is proposed by Gao et al. (2013). An Iterated Greedy (IG) algorithm without local search phases are presented by Lin et al. (2013). A Scatter Search algorithm with a reference set made up of solutions and restarts mechanisms is proposed by Naderi and Ruiz (2014). Fernandez-Viagas and Framinan (2015a) presented an IG algorithm with bounded local search phases employing properties of the problem to reduce the space of solutions. Recently, Ribas et al. (2017) have proposed several constructive heuristics and two simple iterated algorithms (IG algorithm and ILS) with Variable Neighbourhood Searches for the DFSP but with zero-buffer flowshops (blocking constraint).

A particular case of the DFSP refers to the so-called *Distributed Assembly Flowshop Scheduling Problem*, which combines the DFSP with assembly scheduling. In this problem, a distributed flowshop composed of f identical flowshops is followed by a single assembly operation. n jobs consisting each one of f components have to be assembled after each component has been manufactured in one of the flowshops. This decision problem includes job assignment plus the scheduling of jobs in the assembly line. The main references for this problem are Hatami et al. (2015) and Hatami et al. (2013). In the first reference, the authors consider the objective of makespan minimization, while in the second sequence-dependent setup times are assumed. In both cases, the

problem is addressed using approximate algorithms and, as in the assembly scheduling problems, we are not aware of references dealing with stochastic processing times.

Assembly Scheduling

This problem is also denoted *n-stage Assembly* or *Assembly Flowshop Scheduling*. In this problem, m tandem lines are arranged prior to a single assembly station which is fed by the tandem lines. Using this layout, n different products (jobs) have to be manufactured, each one consisting of m components manufactured in the tandem lines. The processing time of each component in each line is different. Some authors distinguish among the *fixed* case (i.e. each component can be processed only in a given tandem line), and the *unfixed* case (i.e. each component can be processed in different factories).

For these problems, different objectives are sought, such as makespan minimization (Sung and Juhn, 2009), total flowtime (Al-Anzi and Allahverdi, 2013; Sung and Kim, 2008), due date fulfilment (Al-Anzi and Allahverdi, 2007), or the combination of several indicators (Seidgar et al., 2014).

Most references refer to the 2-stage case (production followed by assembly), so they assume that each tandem line consists of a single machine. The underlying hypothesis is that there is a single processing time for each component before the assembly process. For this problem, different exact and approximate methods have been proposed, and some variants of the original problem have been tackled by Sung and Juhn (2009), where two types of components –manufactured and imported– are considered, and byLiao et al. (2015), where assembly batches are assumed.

Several other variants of the problem for three stages have been addressed in the literature (see e.g. Koulamas and Kyparisis, 2001 and Komaki et al., 2017), but in none of the different versions of the problem the processing times have been assumed to be stochastic.

4. Proposed approach

To address the different approaches presented in Section 3, we present three algorithms: the ILS$_M$ algorithm considers the deterministic version of the problem, while the SIM-ILS$_{EM}$ and the SIM-ILS$_{MP}$ algorithms minimize the expected makespan and the percentile makespan, respectively. For each solution returned by an algorithm, the (deterministic) makespan, the expected makespan and the makespan percentile are computed. The aim of working with different algorithms is to study and compare their behaviour. While simulation techniques are used in the SIM-ILS$_{EM}$ and the SIM-ILS$_{MP}$ algorithms, the ILS$_M$ algorithm, which works with average processing times, skips that part. From here, we use SIM-ILS algorithm to refer to the basic structure of the SIM-ILS$_{EM}$ and the SIM-ILS$_{MP}$ algorithms.

Currently, there is a lack of methodologies for solving stochastic combinatorial optimization problems such as the one under consideration, and most of the existing ones present drawbacks. Simheuristics represent a powerful alternative, which integrates simulation into metaheuristic-driven frameworks. One drawback that most methodologies face is related to the size of the instances. They are not able to solve large-size instances in reasonable amounts of times, whereas simheuristics are scalable. In fact, simheuristics are able to obtain good solutions very fast because simulation (which may be time-consuming) is only used to assess a subset of the solutions visited. Another drawback is related to the probability distributions of the processing times. Simheuristics do not make any assumption, but some methodologies can only work with specific probability distributions (Juan et al., 2014a). As a consequence, simheuristics have been used in a wide range of fields such as routing, scheduling, manufacturing, and healthcare, among others (Juan et al., 2014a; Gonzalez et al., 2016; De Armas et al., 2016b, among others).

We propose a SIM-ILS algorithm combining the Iterated Local Search (ILS) metaheuristic with Monte Carlo simulation (MCS). The ILS is a simple methodology that generates a sequence of solutions applying iteratively local search to perturb the current search point by using repeated random trials in the space of local optima (Lourenço et al., 2010). As the literature shows, the ILS is a successful framework to solve the PFSP. The IG metaheuristic, which has an ILS-based framework, was first applied to the PFSP considering makespan minimization by Ruiz and Stützle (2007). Its good performance and simple implementation have boosted its application to other scheduling problems (Pan and Ruiz, 2014; Hatami et al., 2015, among others). In our approach, the metaheuristic searches for promising solutions while MCS techniques are employed to assess

their performance. The promising solutions are returned by the metaheuristic when solving a (deterministic) PFSP instance, which is created from the original PFSPST instance by replacing the random variables P_{ij} by constant values p_{ij} using the means, i.e., $p_{ij} = E[P_{ij}]$. Note that a solution for the PFSP is also feasible for the PFSPST. Simulation is applied to a given solution to compute the expected makespan or makespan percentile. This methodology assumes that there is a strong correlation between solutions for the PFSP and the PFSPST but not perfect. In other words, the best solution for the PFSP does not have to be the same that the best for the PFSPST, but good solutions for one problem will tend to have a good performance in both.

Our algorithm works with a list of best stochastic solutions found and the best deterministic solution found. The best deterministic one is the job sequence with the smallest makespan referring to the PFSP instance. Depending on the objective considered, the best stochastic solutions found are the job sequences with the smallest expected makespans or makespan percentiles, referring to the PFSPST instance. The algorithm starts solving the PFSP. The obtained result is set as the best deterministic solution and the best stochastic solution. During the algorithm execution, the best stochastic solutions are saved in a list with length l. This list is sorted iteratively in increasing order of the considered objective function. Thus, the solution at the first position is considered as the best stochastic solution. The steps of our algorithm are detailed in Figure B.3 and explained below.

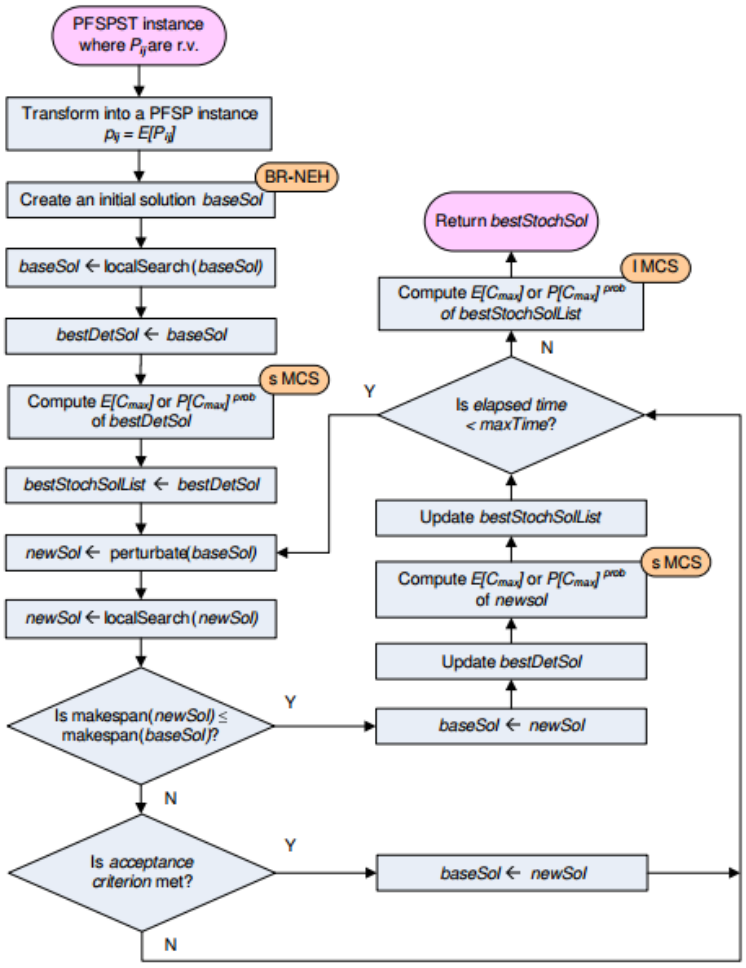


FIGURE B.3: Flowchart of the SIM-ILS algorithm proposed.

Generation of the initial solution

A biased-randomized version of the classical NEH heuristic (Nawaz et al., 1983) described in Juan et al. (2014a) is proposed to generate initial solutions. While the use of random solutions is not unusual, most authors recommend to avoid them (Ruiz and Stützle, 2007; Naderi and Ruiz, 2010;

Vallada and Ruiz, 2010). This randomized version is able to provide different initial solutions when is repeatedly executed using different seeds for the random number generator.

The NEH heuristic is an iterative algorithm with two stages. First, an initial order is generated by sorting the jobs by total completion time on all machines in non-increasing order. In the second stage, jobs are iteratively inserted into a partial sequence according to the order. Jobs are inserted at the position of the partial sequence that results in the minimum makespan. Since all steps are deterministic, there is only one possible solution. In the biased-randomized version, a skewed distribution probability is employed to assign a probability of being selected to each job. Following the logic behind the classical heuristic, the jobs with higher completion times are assigned a higher probability. The discrete version of the decreasing triangular distribution is used. For more details, the reader is referred to Juan et al. (2014a).

Solution improvement

An iterative improvement procedure using *shift-to-left* as first-improvement type pivoting rule is applied in different parts of our algorithm to improve solutions. This procedure has been proposed in several works (Ruiz and Stützle, 2007; Juan et al., 2014a). Each iteration of the procedure consists of three steps. In the first, a position s is randomly selected, without repetition, from the current job sequence. The selected positions are saved in a selection list. In the second step, the job placed in the position s is removed from the sequence and the *shift-to-left* movement is applied, i.e., the insertion of the job in each possible position at the left side of s is tested (see Figure B.4). The makespan of each option is calculated through the accelerations of Taillard (Taillard, 1990). Finally, the job is inserted in the position resulting in the sequence with the smallest makespan. The iteration of these steps are continued until all positions have been selected or a better solution is achieved. If there is an improvement, the algorithm is restarted with an empty selection list.

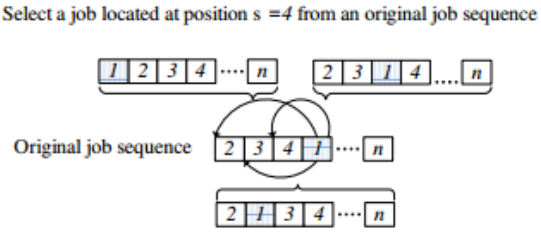


FIGURE B.4: *shift-to-left* movement.

Simulation

The assessment of a solution using MCS techniques follows these steps: *a)* a number of iterations *numsim* is considered to repeat the simulation process, *b)* a job processing time is generated for each random variable according to the probability distribution associated, and the makespan is computed, *c)* this process is repeated *numsim* times, *d)* a performance measure such as the expected makespan, $E[C_{max}]$, or the makespan percentile for a probability *pro*, $P[C_{max}]^{pro}$, is computed. While, the assessment of solutions during the search is done quickly (i.e., *numsim* is relatively small), a long simulation (*numsim* relatively big) is used at the end to provide accurate estimates related to the best deterministic and best stochastic solutions.

Iterated local search

A series of steps are performed iteratively during the search. Initially, a perturbation operator is applied to change the region of the current solution space and then, the new solution is improved using the local search explained in the previous subsection. The simple and efficient *enhanced-swap* operator proposed by Juan et al. (2014f) is used to perturb the solution. It takes three steps: *(1)* two different positions are selected randomly from the current job sequence; *(2)* the jobs at these positions are interchanged; and *(3)* the *shift-to-left* movement is applied for both jobs.

In the second step, the algorithm decides whether the new solution is accepted. If it has a smaller makespan than the current base solution, then the latter is replaced by the new. In this

case, the best deterministic solution is accordingly updated (i.e., replaced by the new solution if this has a smaller makespan). Additionally, a short simulation is applied to check whether the best stochastic solution list has also to be updated considering the objective function value. Finally, if the new solution does not provide a smaller makespan than the current base solution, an acceptance criterion is applied (see next subsection). These steps are repeated until the stopping criterion based on the elapsed CPU time is reached.

Acceptance criteria

Our algorithm assigns an acceptance probability to the new solutions that are worse than the current base solution. This criterion prevents the algorithm from getting stuck in a local optima. It is used for the first time by Hatami et al. (2015) and is based on the probabilistic acceptance criterion of the Simulated Annealing, which has been extensively used (Ruiz and Stützle, 2008; Wang et al., 2015; Yu and Lin, 2015, among others).

Given a new solution π_n with a worse makespan than the current base solution π_C, the acceptance criterion decides if it is accepted or not. Let $C_{Max}(\pi_c)$ and $C_{Max}(\pi_n)$ denote the makespans of each solution. The acceptance of π_n depends on the probabilistic mechanism shown in Equation B.1, where *random* is a random number uniformly distributed between 0 and 1, and *temp* is a parameter (Osman and Potts, 1989).

$$random \leq e^{-\frac{C_{Max}(\pi_n)-C_{Max}(\pi_c)}{temp}}. \tag{B.1}$$

This mechanism is simplified in two aspects by Hatami et al. (2015). First, the parameter is removed since the results in Ruiz and Stützle (2007) and Ruiz and Stützle (2008) prove that its effects are not statistically significant. Second, the difference of makespans (which can be the same for instances with distinct quality) is replaced by the relative percentage difference (RPD): $RPD = \frac{C_{(\pi_n)}-C_{(\pi_c)}}{C_{(\pi_c)}} \times 100$. Thus, this acceptance criteria (Equation B.2) is simpler and avoids the need of parameter fine-tuning.

$$random \leq e^{-RPD}. \tag{B.2}$$

5. Computational experiments

The algorithms described in the previous section have been implemented as Java applications and tested on 27 instances. A standard personal computer, Intel QuadCore i5 CPU at 3.2 GHz and 4 GB RAM with Windows 7, has been used to execute all tests. This section provides the description of the instances, the tests carried out, and the numerical results. The analysis of the results is presented in the next section.

Set of instances and test

Since no benchmark instances exist for the problem analyzed, a new set is constructed based on Taillard instances (Taillard, 1993). Table B.1 gathers the following characteristics for each instance: name, total number of the jobs (total n), number of machines (m) and number of factories (f). For a given factory, each instance contains a processing time p_{ij} for job i at machine j, which describes a random variable P_{ij} following a Log-normal distribution with mean p_{ij} and variance σ_{ij}^2 set to $c \cdot p_{ij}$. In real-life applications, empirical distributions based on historical data could be used. Instances are available from the authors upon request.

Table B.1: Instance description.

	Total n								
	20			50			100		
f / m	5	10	20	5	10	20	5	10	20
2	Ins. 1	Ins. 4	Ins. 7	Ins. 10	Ins. 13	Ins. 16	Ins. 19	Ins. 22	Ins. 25
3	Ins. 2	Ins. 5	Ins. 8	Ins. 11	Ins. 14	Ins. 17	Ins. 20	Ins. 23	Ins. 26
4	Ins. 3	Ins. 6	Ins. 9	Ins. 12	Ins. 15	Ins. 18	Ins. 21	Ins. 24	Ins. 27

Three different levels of processing time variability c (small, medium and high) are considered and set to 0.25, 1 and 1.5, respectively. Three different values of 80%, 90% and 95% are considered for the general probability p (used only for the SIM-ILS$_{MP}$ algorithm). The maximum computational time for solving the PFSPST of each factory is limited to $0.05 \cdot n \cdot m$, which seems a reasonable amount for real-life applications. Ten seeds are randomly generated and only the best result is stored. Regarding the number of iterations for assessing solutions, 600 and 1000 runs are employed during the algorithm and at the end, respectively. Note that the selection of these values are mainly driven by the computing time available. Thus, if more time is available, then these values can be incremented in order to obtain better and more accurate results.

Results

Results are displayed in Tables B.2-B.4, where each table represents a specific level of processing time variability: low, medium and high. Due to space limitations and the fact that results show similar trends for all three values of general probability, only those related to 90% are shown. The composition of the tables is as follows. The first column identifies the instance. The next five summarize the results of the ILS$_M$ algorithm, which considers makespan minimization. For each instance, they show the following information regarding the best solution found: $C_{max}(1)$, $E[C_{max}](2)$, $P[C_{max}]^{pro}(3)$, gap between the first two measures, computed as: $(E[C_{max}](2) - C_{max}(1))/C_{max}(1) \cdot 100$, and gap between the second and the third ones. While the first gap represents the 'extra' processing time, on average, for applying a solution assuming deterministic processing times, the second focuses on percentiles, showing the additional processing time required to finish the product with a probability of 90% (note that this time could be negative). The next four columns provide the following results of the SIM-ILS$_{EM}$ algorithm, which minimizes the expected makespan: $E[C_{max}](4)$, $P[C_{max}]^{pro}(5)$, gap between the expected makespan of the best solutions found by the ILS$_M$ and the SIM-ILS$_{EM}$ algorithms, and the gap of percentiles among the same solutions. The third gap, which is expected to be null or negative, shows the benefit of using a simheuristic approach (i.e., taking into account the variability of the processing times) in terms of expected makespan. The fourth gap quantifies the difference of percentiles. Similarly, the next four columns refer to the best solution found by the SIM-ILS$_{MP}$ algorithm, which minimizes the makespan percentile. In particular, they contain: $E[C_{max}](6)$, $P[C_{max}]^{pro}(7)$, and gaps of expected makespans and percentiles between the best solutions found by the SIM-ILS$_{EM}$ and SIM-ILS$_{MP}$ algorithms. These gaps allow us to quantify the processing time difference based on whether we minimize one measure or the other. Finally, the last column shows the mean computational time of the three solutions obtained. In addition, a row is added at the end of each table to gather the mean gaps and computational time among instances.

Boxplots in Figure B.5 show the distributions of gaps of $E[C_{max}]$ and $P[C_{max}]^{pro}$ regarding the best values considering the three algorithms and a probability of 90%. While we expect that the approach minimizing a given measure present a null value for the corresponding gap, this figure reveals the difference between choosing one approach or the other, allowing us to analyse the variability associated to these gaps. Focusing on the instance 14, FigureB.6 represents the 30 solutions found (resulting of 3 algorithms and 10 seeds). Each column is a measure, and colors and line formats are used to distinguish algorithms. As the previous figure, this analysis provides insights about a "potential" trade-off between the measures. Additionally, this figure gives information about the effect of using multiple seeds.

Figure B.7 represents the relationship between probability required, variability level of the processing times and $P[C_{max}]^{pro}$ for the instance 14 using the SIM-ILS$_{MP}$. Finally, Figure B.8 shows the effects of different instance characteristics on $P[C_{max}]^{pro}$ considering a medium level of variability and a probability of 90%. First, an analysis of variance was carried out to identify which factors and pairwise interactions had a statistically significant effect on the results. For each of these elements (single factors or pair of them), a figure is drawn which shows the mean value associated to each level of the factor or combination of levels for pair of factors. Given the randomness in the generation of instances, we expect that all factors have significant positive effects.

TABLE B.2: Results considering low level of variability (c = 0.25) and general probability p = 90%.

	ILS$_M$					SIM-ILS$_{EM}$				SIM-ILS$_{MP}$				
Instance	C_{max} (1)	$E[C_{max}]$ (2)	$P[C_{max}]_{PRO}$ (3)	Gap (2-1) (%)	Gap (3-2) (%)	$E[C_{max}]$ (4)	$P[C_{max}]_{PRO}$ (5)	Gap (4-2) (%)	Gap (5-3) (%)	$E[C_{max}]$ (6)	$P[C_{max}]_{PRO}$ (7)	Gap (6-4) (%)	Gap (7-5) (%)	Mean CPU time
1	1505	1516.00	1555.54	0.73	2.61	1512.33	1552.96	-0.24	-0.17	1514.19	1551.75	0.12	-0.08	5.00
2	1758	1772.30	1829.89	0.81	3.25	1762.47	1824.72	-0.55	-0.28	1765.96	1822.13	0.20	-0.14	3.99
3	1862	1869.94	1949.62	0.43	4.26	1864.56	1944.33	-0.29	-0.27	1865.41	1942.73	0.05	-0.08	4.50
4	2154	2169.84	2216.99	0.74	2.17	2167.72	2213.02	-0.10	-0.18	2169.41	2212.05	0.08	-0.04	9.99
5	2829	2854.76	2927.28	0.91	2.54	2852.35	2923.38	-0.08	-0.13	2854.23	2921.13	0.07	-0.08	10.00
6	3179	3194.56	3295.33	0.49	3.15	3189.44	3294.94	-0.16	-0.01	3191.61	3288.55	0.07	-0.19	7.99
7	3413	3451.80	3503.35	1.14	1.49	3440.75	3495.76	-0.32	-0.22	3442.72	3493.16	0.06	-0.07	19.99
8	4306	4333.25	4421.54	0.63	2.04	4332.04	4424.23	-0.03	0.06	4334.49	4418.29	0.06	-0.13	20.00
9	5733	5760.93	5904.21	0.49	2.49	5752.36	5890.05	-0.15	-0.24	5756.43	5884.47	0.07	-0.09	14.00
10	2960	2987.24	3040.81	0.92	1.79	2960.68	3020.68	-0.89	-0.66	2964.55	3019.89	0.13	-0.03	12.46
11	3250	3285.55	3352.58	1.09	2.04	3272.02	3349.40	-0.41	-0.09	3273.94	3346.25	0.06	-0.09	12.49
12	3378	3417.10	3505.31	1.16	2.58	3392.12	3501.08	-0.73	-0.12	3395.54	3490.67	0.10	-0.30	12.49
13	3572	3620.75	3669.11	1.36	1.34	3620.75	3669.11	0.00	0.00	3620.75	3669.11	0.00	0.00	25.01
14	4031	4093.01	4172.92	1.54	1.95	4083.66	4163.44	-0.23	-0.23	4084.43	4156.70	0.02	-0.16	25.00
15	4574	4620.28	4728.14	1.01	2.33	4605.65	4720.76	-0.32	-0.16	4609.00	4714.98	0.07	-0.12	24.99
16	5058	5132.22	5194.14	1.47	1.21	5131.72	5191.09	-0.01	-0.06	5131.72	5191.09	0.00	0.00	49.99
17	6039	6113.22	6205.43	1.23	1.51	6109.34	6208.90	-0.06	0.06	6113.31	6204.41	0.06	-0.07	49.97
18	7013	7090.64	7215.40	1.11	1.76	7081.97	7215.77	-0.12	0.01	7085.22	7204.73	0.05	-0.15	49.98
19	5797	5840.48	5913.77	0.75	1.25	5810.07	5891.05	-0.52	-0.38	5818.16	5889.75	0.14	-0.02	24.98
20	5819	5854.95	5967.56	0.62	1.92	5825.96	5939.00	-0.50	-0.48	5832.76	5935.59	0.12	-0.06	24.95
21	6065	6104.18	6236.87	0.65	2.17	6078.89	6220.23	-0.41	-0.27	6083.24	6217.93	0.07	-0.04	24.98
22	6235	6324.25	6392.90	1.43	1.09	6324.25	6392.90	0.00	0.00	6326.47	6388.82	0.04	-0.06	49.98
23	6414	6502.24	6598.61	1.38	1.48	6498.90	6592.36	-0.05	-0.09	6498.90	6592.36	0.00	0.00	50.01
24	7183	7289.80	7422.74	1.49	1.82	7283.76	7418.55	-0.08	-0.06	7283.75	7407.78	0.00	-0.15	50.00
25	7603	7697.50	7763.14	1.24	0.85	7697.50	7763.14	0.00	0.00	7697.50	7763.14	0.00	0.00	100.03
26	8598	8710.46	8823.77	1.31	1.30	8710.46	8823.77	0.00	0.00	8710.46	8823.77	0.00	0.00	100.01
27	9892	10038.66	10178.40	1.48	1.39	10029.92	10175.82	-0.09	-0.03	10029.92	10175.82	0.00	0.00	99.99
Mean				1.02	1.99			-0.24	-0.15			0.06	-0.08	32.69

TABLE B.3: Results considering medium level of variability ($c = 1$) and general probability $p = 90\%$.

		ILS$_M$				SIM-ILS$_{EM}$				SIM-ILS$_{MP}$			
Instance	C_{max} (1)	$E[C_{max}]$ (2)	$P[C_{max}]^{pro}$ (3)	Gap (2-1) (%)	Gap (3-2) (%)	$E[C_{max}]$ (4)	$P[C_{max}]^{pro}$ (5)	Gap (4-2) (%)	Gap (5-3) (%)	$E[C_{max}]$ (6)	$P[C_{max}]^{pro}$ (7)	Gap (6-4) (%)	Gap (7-5) (%)	Mean CPU time
1	1505	1541.15	1617.25	2.40	4.94	1533.01	1614.37	-0.53	-0.18	1533.96	1606.78	0.06	-0.47	5.00
2	1758	1799.07	1927.61	2.34	7.14	1782.58	1901.69	-0.92	-1.34	1787.49	1897.95	0.28	-0.20	3.99
3	1862	1880.42	2046.20	0.99	8.82	1876.25	2041.74	-0.22	-0.22	1879.09	2030.26	0.15	-0.56	4.50
4	2154	2208.99	2298.54	2.55	4.05	2203.58	2295.21	-0.25	-0.14	2205.94	2287.00	0.11	-0.36	10.00
5	2829	2898.36	3035.07	2.45	4.72	2896.75	3033.22	-0.06	-0.06	2899.76	3032.76	0.10	-0.02	9.99
6	3179	3223.10	3424.39	1.39	6.25	3215.97	3414.35	-0.22	-0.29	3215.97	3414.35	0.00	0.00	8.00
7	3413	3517.97	3617.33	3.08	2.82	3493.77	3603.77	-0.69	-0.37	3495.26	3594.01	0.04	-0.27	19.99
8	4306	4387.69	4552.74	1.90	3.76	4380.10	4564.70	-0.17	0.26	4382.97	4552.50	0.07	-0.27	19.99
9	5733	5811.38	6094.35	1.37	4.87	5792.84	6067.04	-0.32	-0.45	5797.46	6055.08	0.08	-0.20	14.00
10	2960	3030.64	3136.25	2.39	3.48	2979.60	3094.51	-1.68	-1.33	2983.69	3090.15	0.14	-0.14	12.49
11	3250	3336.39	3502.73	2.66	4.99	3321.23	3475.18	-0.45	-0.79	3326.65	3466.45	0.16	-0.25	12.49
12	3378	3482.60	3673.45	3.10	5.48	3431.44	3638.61	-1.47	-0.95	3435.06	3627.96	0.11	-0.29	12.51
13	3573	3710.11	3810.50	3.84	2.71	3708.59	3810.36	-0.04	0.00	3708.59	3810.36	0.00	0.00	25.00
14	4031	4181.10	4332.06	3.72	3.61	4162.39	4322.68	-0.45	-0.22	4174.02	4317.41	0.28	-0.12	25.01
15	4574	4704.41	4911.53	2.85	4.40	4683.63	4901.01	-0.44	-0.21	4686.14	4896.47	0.05	-0.09	25.00
16	5058	5249.31	5356.33	3.78	2.04	5243.28	5349.75	-0.11	-0.12	5243.28	5349.75	0.00	0.00	50.00
17	6039	6241.34	6419.70	3.35	2.86	6226.07	6409.99	-0.24	-0.15	6229.50	6405.71	0.05	-0.07	49.98
18	7013	7205.94	7440.63	2.75	3.26	7205.49	7448.99	-0.01	0.11	7205.94	7440.63	0.01	-0.11	49.98
19	5797	5897.76	6042.62	1.74	2.46	5855.19	6010.80	-0.72	-0.53	5866.03	6001.97	0.19	-0.15	24.96
20	5819	5954.39	6150.09	2.33	3.29	5874.48	6086.12	-1.34	-1.04	5887.17	6081.12	0.22	-0.08	24.92
21	6065	6181.98	6426.37	1.93	3.95	6128.44	6403.68	-0.87	-0.35	6144.92	6395.87	0.27	-0.12	24.99
22	6233	6449.74	6575.00	3.48	1.94	6441.78	6569.46	-0.12	-0.08	6441.78	6569.46	0.00	0.00	50.00
23	6415	6663.16	6858.39	3.87	2.93	6633.10	6821.64	-0.45	-0.54	6642.01	6820.06	0.13	-0.02	49.99
24	7183	7464.61	7726.46	3.92	3.51	7428.87	7676.50	-0.48	-0.65	7440.15	7674.73	0.15	-0.02	49.98
25	7600	7863.12	7990.74	3.46	1.62	7863.12	7990.74	0.00	0.00	7863.12	7990.74	0.00	0.00	100.00
26	8592	8897.85	9103.21	3.56	2.31	8897.85	9103.21	0.00	0.00	8897.85	9103.21	0.00	0.00	99.98
27	9888	10267.27	10539.56	3.84	2.65	10250.60	10535.72	-0.16	-0.04	10252.56	10529.88	0.02	-0.06	99.97
Mean				2.78	3.88			-0.46	-0.36			0.10	-0.14	32.69

TABLE B.4: Results considering high level of variability ($c = 1.5$) and general probability $p = 90\%$.

		ILS$_M$			SIM-ILS$_{EM}$				SIM-ILS$_{MP}$				
Instance	C_{max} (1)	$E[C_{max}]$ (2)	$P[C_{max}]^{pro}$ (3)	Gap (2-1) (%)	Gap (3-2) (%)	$E[C_{max}]$ (4)	$P[C_{max}]^{pro}$ (5)	Gap (4-2) (%)	Gap (5-3) (%)	$E[C_{max}]$ (6)	$P[C_{max}]^{pro}$ (7)	Gap (6-4) (%)	Gap (7-5) (%)	Mean CPU time
1	1505	1558.65	1652.71	3.56	6.03	1543.93	1645.57	-0.94	-0.43	1547.66	1634.96	0.24	-0.64	5.00
2	1758	1815.86	1971.38	3.29	8.56	1793.26	1936.85	-1.24	-1.75	1796.58	1932.27	0.19	-0.24	3.99
3	1862	1899.64	2088.00	2.02	9.92	1883.39	2087.17	-0.86	-0.04	1887.72	2070.03	0.23	-0.82	4.50
4	2154	2230.49	2338.89	3.55	4.86	2223.55	2331.53	-0.31	-0.31	2225.38	2325.86	0.08	-0.24	10.00
5	2829	2923.50	3100.37	3.34	6.05	2920.98	3095.04	-0.09	-0.17	2925.05	3089.05	0.14	-0.19	9.99
6	3179	3250.82	3500.13	2.26	7.67	3231.39	3488.72	-0.60	-0.33	3239.13	3474.39	0.24	-0.41	8.00
7	3413	3550.25	3693.79	4.02	4.04	3521.54	3653.49	-0.81	-1.09	3522.88	3647.18	0.04	-0.17	20.00
8	4306	4417.80	4645.59	2.60	5.16	4408.96	4633.95	-0.20	-0.25	4414.71	4618.62	0.13	-0.33	19.99
9	5733	5829.65	6148.97	1.69	5.48	5815.95	6148.41	-0.23	-0.01	5820.54	6141.05	0.08	-0.12	14.00
10	2960	3051.56	3186.60	3.09	4.43	2989.60	3128.79	-2.03	-1.81	2989.60	3128.79	0.00	0.00	12.46
11	3250	3369.09	3547.77	3.66	5.30	3348.04	3530.25	-0.62	-0.49	3354.05	3526.75	0.18	-0.10	12.50
12	3378	3510.02	3741.10	3.91	6.58	3455.00	3711.78	-1.57	-0.78	3459.46	3691.02	0.13	-0.56	12.49
13	3570	3745.06	3862.43	4.90	3.13	3745.06	3862.43	0.00	0.00	3745.06	3862.43	0.00	0.00	24.99
14	4031	4221.27	4401.91	4.72	4.28	4206.78	4396.52	-0.34	-0.12	4210.35	4395.90	0.08	-0.01	24.99
15	4574	4743.71	4997.22	3.71	5.34	4729.26	5001.05	-0.30	0.08	4737.43	4987.14	0.17	-0.28	24.99
16	5058	5312.82	5448.19	5.04	2.55	5304.92	5443.07	-0.15	-0.09	5304.92	5443.07	0.00	0.00	50.01
17	6039	6301.25	6533.04	4.34	3.68	6289.15	6517.20	-0.19	-0.24	6293.81	6507.38	0.07	-0.15	49.97
18	7013	7282.57	7597.07	3.84	4.32	7273.14	7594.89	-0.13	-0.03	7277.18	7565.94	0.06	-0.38	49.99
19	5797	5967.38	6140.62	2.94	2.90	5881.63	6066.46	-1.44	-1.21	5889.21	6054.73	0.13	-0.19	24.99
20	5819	5990.94	6236.24	2.95	4.09	5908.93	6161.28	-1.37	-1.20	5916.77	6155.52	0.13	-0.09	24.94
21	6065	6230.30	6563.94	2.73	5.36	6169.34	6505.67	-0.98	-0.89	6172.18	6485.97	0.05	-0.30	24.98
22	6237	6531.98	6675.40	4.73	2.20	6509.74	6672.00	-0.34	-0.05	6510.28	6662.21	0.01	-0.15	49.99
23	6411	6718.36	6953.37	4.79	3.50	6707.55	6922.70	-0.16	-0.44	6707.55	6922.70	0.00	0.00	50.00
24	7183	7519.80	7820.43	4.69	4.00	7499.62	7802.18	-0.27	-0.23	7499.62	7802.18	0.00	0.00	50.00
25	7603	7970.91	8150.20	4.84	2.25	7962.01	8126.37	-0.11	-0.29	7962.01	8126.37	0.00	0.00	100.00
26	8600	9011.27	9264.51	4.78	2.81	9011.27	9264.51	0.00	0.00	9011.27	9264.51	0.00	0.00	99.98
27	9892	10386.80	10731.39	5.00	3.32	10354.76	10707.00	-0.31	-0.23	10354.76	10707.00	0.00	0.00	99.99
Mean				3.74	4.73			-0.58	-0.46			0.09	-0.20	32.69

6. Analysis of results

Tables B.2-B.4 provide detailed information on the performance of our algorithms. The following comments refer to the results of the ILS_M algorithm. Mean gaps between C_{max} and $E[C_{max}]$ for small, medium and high levels of variability are 1.02%, 2.78%, and 3.74%, respectively. These values between $E[C_{max}]$ and $P[C_{max}]^{pro}$ are 1.99%, 3.88%, and 4.73%. These values quantify the extra processing time required, on average, when variability is not considered, and the processing time needed to satisfy the deadline with a probability of 90%. For example, in the scenario of low variability, the processing time will be on average 1.02% higher than that assumed, and the processing time needed to finish with the specific probability will be 1.99% higher than the $E[C_{max}]$. Both gaps increase as the variability is incremented. Comparing the results of the ILS_M and the SIM-ILS_{EM} algorithms, the mean gaps of $E[C_{max}]$ (−0.24%, −0.46% and −0.58%) and $P[SC_{max}]^{pro}$ (−0.15%, −0.36% and −0.46%) quantify the benefits of using a simheuristic algorithm. Regarding the results of the SIM-ILS_{EM} and SIM-ILS_{MP} algorithms, the mean gaps of $E[C_{max}]$ (0.06%, 0.10% and 0.09%) and $P[C_{max}]^{pro}$ (−0.08%, −0.14% and −0.20%) at different level of variability, evidence the benefits of using one or the other approach. Thus, the main findings are: *(i)* ignoring the variability in processing times may have an important effect on the performance measures (even in scenarios with a low level of variability); *(ii)* the solutions found by the SIM-ILS_{EM} and the SIM-ILS_{MP} algorithms are relatively similar in terms of these measures but not equal; and *(iii)* the gaps tend to increase with the variability, i.e., minimizing the expected makespan is almost equivalent to consider the makespan percentile when the variability is low, but the difference increases as the variability is incremented. As a consequence, a decision-maker have to assess whether he prefers to minimize the expected makespan (i.e., processing finished at the deadline, on average) or the percentile (i.e., be sure that the processing will be finished at the deadline or before with a given probability), which may be seen as a more conservative or risk-aversion approach.

Figure B.5 compares performance measures among the algorithms proposed. The distributions of the gaps are relatively symmetric, with few outliers on right tails. It is easy to see that the biggest gaps are related to the ILS_M algorithm, while the gaps referring to $P[SC_{max}]^{pro}$ values are higher. Similarly, Figure B.6 shows that there is a stronger correlation between the simheuristic-based algorithms in the sense that the profiles are similar. It is interesting to analyse the differences among the solutions obtained with multiple seeds. For instance, while solutions of the ILS_M algorithm have the same or a similar C_{max}, these solutions may differ significantly in the other measures (i.e., there are solutions more robust than others). For the instance studied, the ranges of the last two measures are higher than that of the first.

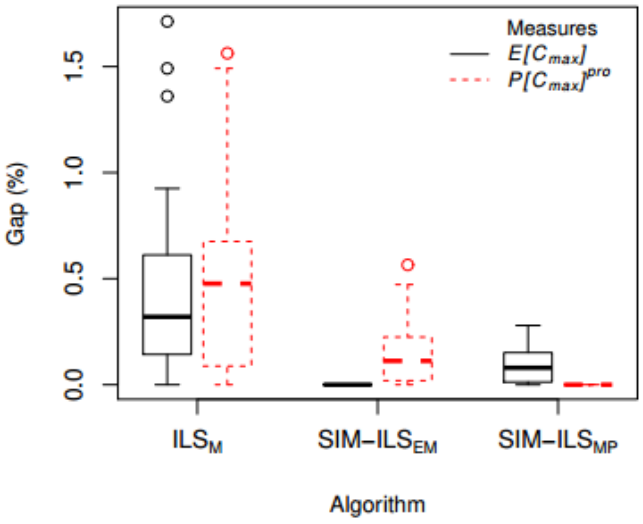


Figure B.5: Boxplots of gaps between C_{max}, $E[C_{max}]$ and $P[C_{max}]^{pro}$ among algorithms considering all instances, medium level of variability and $p = 90\%$.

Figure B.7 represents a valuable tool for a decision-maker. It analyses the relationship between the probability required to process a product at the deadline or before and the processing time

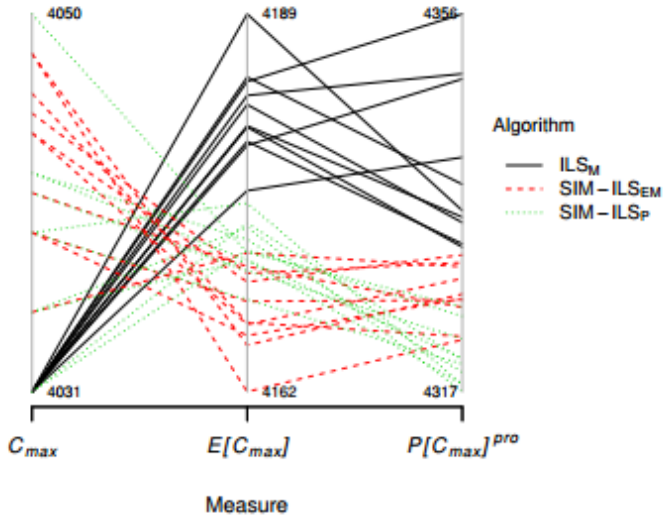


FIGURE B.6: Parallel coordinates plot showing different measures for solutions found with the different algorithms for instance 14, considering a medium level of variability and 10 seeds.

needed. As the probability tends to 1 (i.e., no risk) the processing time tends to infinite. Instead of having a single solution, the decision-maker may choose the best option (given risk-aversion, company policies/situation, etc.) among many. As expected, for a given p value, $P[C_{max}]^{pro}$ increases as the variability is incremented.

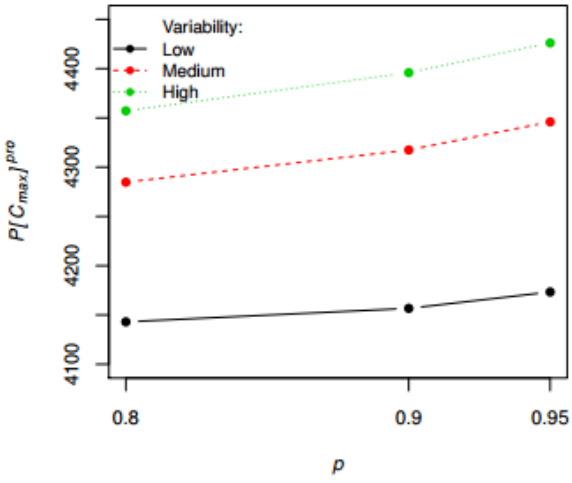


FIGURE B.7: $P[SC_{max}]^{pro}$ as function of general probability and variability level for instance 14 considering the SIM-ILS$_{MP}$ algorithm.

Figure B.8 reveals that factors total n, m, f, and the interaction between f and m have statistically significant and positive effects (when considering the others elements) on $P[C_{max}]^{pro}$. The ranges related to total n and m are the highest. While the effects of f and total n seem lineal, the effect of m draws a convex function. Focusing on the interaction, it can be concluded that the effect of f is positive for any value of m, but $P[C_{max}]^{pro}$ increases as m is incremented.

7. Conclusions

The manufacturing industry is becoming increasingly complex and competitive. Companies need powerful optimization algorithms to design proper strategies that make them efficient in order to remain in the market. Although there is an extensive literature on classical scheduling problems,

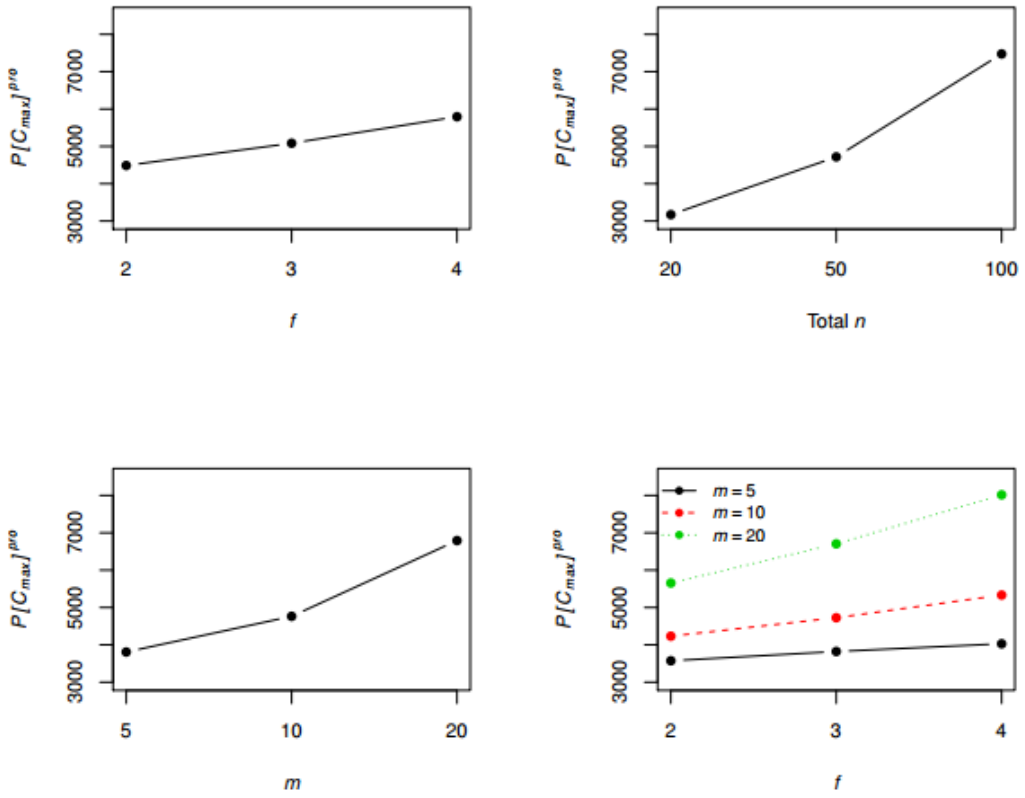


FIGURE B.8: Effect of different instance characteristics on $P[C_{max}]^{pro}$ considering the SIM-ILS$_{MP}$ algorithm.

there is a lack of works on richer and more realistic problems. In this context, our work studies a novel problem called Distributed Permutation Flowshop Scheduling Problem with Stochastic processing Times. It consists in the manufacturing of a product that requires several jobs that are performed in independent factories. The sub-problem of each factory can be modelled as a Permutation Flowshop Scheduling Problem with Stochastic processing Times. All factories are expected to finish at a given deadline or before. This problem describes several real-life applications where a company acquires intermediate products from others and assembles them to obtain a final product with a higher added value.

Three algorithms are proposed to deal with this problem which aim to minimize a different objective function: the makespan (ignoring the stochasticity), the expected makespan and the makespan percentile given a probability p. This percentile is the value below which a given proportion p of makespans fall when simulating scenarios, and can be interpreted as follow: if the starting time in a factory is set to the deadline minus this percentile, the processing of the product will be finished before or at the deadline with a probability p. While all algorithms rely on the Iterated Local Search metaheuristic, the second and the third ones are simheuristic algorithms, i.e., integrate Monte Carlo simulation techniques in order to deal with the stochasticity. Note that the second algorithm is intended to provide good results on average whereas the third one aims to guarantee that the manufacturing will be finished before or at the deadline with a given probability. A set of computational experiments allow us to compare the algorithms in terms of makespan, expected makespan and makespan percentile, and quantify these differences. It is proven that: *(i)* gaps among algorithms for each measure increase as the level of stochasticity is incremented; *(ii)* while there is a strong correlation between simheuristic algorithms (in the sense that solutions having the best performance in terms of expected makespan are also of good quality regarding makespan percentile, and the other way around), it is weaker between the first algorithm and any of the others; *(iii)* in some cases the differences between the second and the third algorithm may be significant, so a priority must be set by the decision-maker; *(iv)* the fact that the algorithms are so fast enable the running of the third one considering different probabilities, which provides a deeper insight of the relationship between probability (related to the risk-aversion, i.e., how sure

decision-maker wants to be about finishing at a given deadline or before) and makespan percentile (i.e., how much time he needs to start before the deadline); *(v)* the effect of using different seeds is significant; and *(vi)* the makespan percentile linearly depends on the number of factories, jobs and machines, and the interaction between number of factories and machines.

Acknowledgments

This work has been partially supported by the Spanish Ministry of Economy and Competitiveness and FEDER (TRA2015-71883-REDT, CYTED2014-P514RT0013, DPI2013-44461-P, DPI2016-80750-P). Likewise, we want to thank the support of the UOC doctoral programme.

References

Al-Anzi, F. and A. Allahverdi (2007). "A self-adaptive differential evolution heuristic for two-stage assembly scheduling problem to minimize maximum lateness with setup times". In: *European Journal of Operational Research* 182.1, pp. 80–94.

Al-Anzi, F. and A. Allahverdi (2013). "An artificial immune system heuristic for two-stage multi-machine assembly scheduling problem to minimize total completion time". In: *Journal of Manufacturing Systems* 32.4, pp. 825–830.

Al Kattan, I. and R. Maragoud (2008). "Performance analysis of flowshop scheduling using genetic algorithm enhanced with simulation". In: *International Journal of Industrial Engineering: Theory, Applications and Practice* 15.1, pp. 62–72.

Al-Salem, A. (2004). "A heuristic to minimize makespan in proportional parallel flow shops". In: *International Journal of Computing & Information Sciences* 2.2, pp. 98–107.

Allaoui, H., S. Lamouri, and M. Lebbar (2006). "A robustness framework for a stochastic hybrid flow shop to minimize the makespan". In: *International Conference on Service Systems and Service Management*. Vol. 2. IEEE, pp. 1097–1102.

Baker, K. R. and D. Altheimer (2012). "Heuristic solution methods for the stochastic flow shop problem". In: *European Journal of Operational Research* 216.1, pp. 172–177.

Baker, K. R. and D. Trietsch (2011). "Three heuristic procedures for the stochastic, two-machine flow shop problem". In: *Journal of Scheduling* 14.5, pp. 445–454.

Baker, Kenneth R (1974). *Introduction to sequencing and scheduling*. John Wiley & Sons.

Campbell, H. G., R. A. Dudek, and M. L. Smith (1970). "A heuristic algorithm for the n job, m machine sequencing problem". In: *Management science* 16.10, B630–B637.

Cao, D. and M. Chen (2003). "Parallel flowshop scheduling using Tabu search". In: *International Journal of Production Research* 41.13, pp. 3059–3073.

Chan, F. T.S., S.H. Chung, and P.L.Y. Chan (2005). "An adaptive genetic algorithm with dominated genes for distributed scheduling problems". In: *Expert Systems with Applications* 29.2, pp. 364–371.

Choi, S. H. and K. Wang (2012). "Flexible flow shop scheduling with stochastic processing times: a decomposition-based approach". In: *Computers & Industrial Engineering* 63.2, pp. 362–373.

David, W.H.E., A. Kusiak, and A. Artiba (1996). "A scheduling problem in glass manufacturing". In: *IIE Transactions (Institute of Industrial Engineers)* 28.2, pp. 129–139.

De Armas, J., A. A. Juan, J. M. Marquès, and J. P. Pedroso (2016b). "Solving the deterministic and stochastic uncapacitated facility location problem: from a heuristic to a simheuristic". In: *Journal of the Operational Research Society*.

Fernandez-Viagas, V. and J. M. Framinan (2015a). "A bounded-search iterated greedy algorithm for the distributed permutation flowshop scheduling problem". In: *International Journal of Production Research* 53.4, pp. 1111–1123.

— (2015b). "Efficient non-population-based algorithms for the permutation flowshop scheduling problem with makespan minimisation subject to a maximum tardiness". In: *Computers & Operations Research* 64, pp. 86–96.

— (2015c). "NEH-based heuristics for the permutation flowshop scheduling problem to minimise total tardiness". In: *Computers & Operations Research* 60, 27—36.

Framinan, J. M. and P. Perez-Gonzalez (2015). "On heuristic solutions for the stochastic flowshop scheduling problem". In: *Journal of Operational Research* 246.2, 413—420.

Gao, J. and R. Chen (2011). "A hybrid genetic algorithm for the distributed permutation flowshop scheduling problem". In: *International Journal of Computational Intelligence Systems* 4.4, pp. 497–508.

Gao, J., R. Chen, and W. Deng (2013). "An efficient tabu search algorithm for the distributed permutation flowshop scheduling problem". In: *International Journal of Production Research* 51.3, pp. 641–651.

Garey, M. R., D. S. Johnson, and R. Sethi (1976). "The complexity of flowshop and jobshop scheduling". In: *Mathematics of Operations Research* 1.2, pp. 117–129.

Ghezail, F., H. Pierreval, and S. Hajri-Gabouj (2010). "Analysis of robustness in proactive scheduling: a graphical approach". In: *Computers & Industrial Engineering* 58.2, pp. 193–198.

Hatami, S., R. Ruiz, and C. Andrés-Romano (2013). "The distributed assembly permutation flowshop scheduling problem". In: *International Journal of Production Research* 51.17, pp. 5292–5308.

— (2015). "Heuristics and metaheuristics for the distributed assembly permutation flowshop scheduling problem with sequence dependent setup times". In: *International Journal of Production Economics* 169, pp. 76–88.

Hsu, C.-Y., P.-C. Chang, and M.-H. Chen (2015). "A linkage mining in block-based evolutionary algorithm for permutation flowshop scheduling problem". In: *Computers & Industrial Engineering* 83, pp. 159–171.

Johnson, S.M. (1954). "Optimal two- and three-stage production schedules with setup times included". In: *Naval research logistics quarterly* 1, pp. 61–68.

Kahn, K. B., S. E. Kay, R. J. Slotegraaf, and S. Uban (2013). *The PDMA handbook of new product development*. Third. John Wiley & Sons, Inc. ISBN: 9780470648209.

Katragjini, K., E. Vallada, and R. Ruiz (2013). "Flow shop rescheduling under different types of disruption". In: *International Journal of Production Research* 51.3, pp. 780–797.

Kianfar, K., S. M. T. F. Ghomi, and A. O. Jadid (2012). "Study of stochastic sequence-dependent flexible flow shop via developing a dispatching rule and a hybrid GA". In: *Engineering Applications of Artificial Intelligence* 25.3, pp. 494–506.

Komaki, G.M., E. Teymourian, V. Kayvanfar, and Z. Booyavi (2017). "Improved discrete cuckoo optimization algorithm for the three-stage assembly flowshop scheduling problem". In: *Computers and Industrial Engineering* 105, pp. 158–173.

Koulamas, C. and G. J. Kyparisis (2001). "The three-stage assembly flowshop scheduling problem". In: *Computers & Operations Research* 28.7, pp. 689–704.

Liao, C.-J., C.-H. Lee, and H.-C. Lee (2015). "An efficient heuristic for a two-stage assembly scheduling problem with batch setup times to minimize makespan". In: *Computers & Industrial Engineering* 88, pp. 317–325.

Lin, Q., L. Gao, X. Li, and C. Zhang (2015). "A hybrid backtracking search algorithm for permutation flow-shop scheduling problem". In: *Computers & Industrial Engineering* 85, pp. 437–446.

Lin, S. W., K. C. Ying, and C. Y. Huang (2013). "Minimising makespan in distributed permutation flowshops using a modified iterated greedy algorithm". In: *International Journal of Production Research* 51.16, pp. 5029–5038.

Liu, Q., S. Ullah, and C. Zhang (2011). "An improved genetic algorithm for robust permutation flowshop scheduling". In: *The International Journal of Advanced Manufacturing Technology* 56.1-4, pp. 345–354.

Moon, C., J. Kim, and S. Hur (2002). "Integrated process planning and scheduling with minimizing total tardiness in multi-plants supply chain". In: *Computers & Industrial Engineering* 43.1, pp. 331–349.

Naderi, B. and R. Ruiz (2010). "The distributed permutation flowshop scheduling problem". In: *Computers & Operations Research* 37.4, pp. 754–768.

— (2014). "A scatter search algorithm for the distributed permutation flowshop scheduling problem". In: *European Journal of Operational Research* 239.2, pp. 323–334.

Nawaz, M., J. Enscore, and I. Ham (1983). "A heuristic algorithm for the m-machine, n-job flowshop sequencing problem". In: *Omega* 11, 91—95.

Osman, I. H. and C. N. Potts (1989). "Simulated annealing for permutation flow-shop scheduling". In: *Omega* 17.6, pp. 551–557.

Pan, Q.-K. and R. Ruiz (2014). "An effective iterated greedy algorithm for the mixed no-idle permutation flowshop scheduling problem". In: *Omega* 44, pp. 41–50.

Ribas, I., R. Companys, and X. Tort-Martorell (2017). "Efficient heuristics for the parallel blocking flow shop scheduling problem". In: *Expert Systems with Applications* 74, pp. 41–54.

Rodammer, F. A. and K. P. White (1988). "A recent survey of production scheduling". In: *IEEE Transactions on Systems, Man and Cybernetics* 18.6, pp. 841–851.

Roy, B. (2010). "Robustness in operational research and decision aiding: a multi-faceted issue". In: *European Journal of Operational Research* 200.3, pp. 629–638.

Ruiz, R. and T. Stützle (2007). "A simple and effective iterated greedy algorithm for the permutation flowshop scheduling problem". In: *European Journal of Operational Research* 177.3, pp. 2033–2049.

— (2008). "An iterated greedy heuristic for the sequence dependent setup times flowshop problem with makespan and weighted tardiness objectives". In: *European Journal of Operational Research* 187.3, pp. 1143–1159.

Seidgar, H., M. Kiani, M. Abedi, and H. Fazlollahtabar (2014). "An efficient imperialist competitive algorithm for scheduling in the two-stage assembly flow shop problem". In: *International Journal of Production Research* 52.4, pp. 1240–1256.

Sluga, A., P. Butala, and G. Bervar (1998). "A multi-agent approach to process planning and fabrication in distributed manufacturing". In: *Computers & Industrial Engineering* 35.3, pp. 455–458.

Sung, C. S. and J. Juhn (2009). "Makespan minimization for a 2-stage assembly scheduling problem subject to component available time constraint". In: *International Journal of Production Economics* 119.2, pp. 392–401.

Sung, C. S. and H. A. Kim (2008). "A two-stage multiple-machine assembly scheduling problem for minimizing sum of completion times". In: *International Journal of Production Economics* 113.2, pp. 1038–1048.

Taillard, E. (1990). "Some efficient heuristic methods for the flow shop sequencing problem". In: *European Journal of Operational Research* 47.1, pp. 65–74.

— (1993). "Benchmarks for basic scheduling problems". In: *European Journal of Operational Research* 64, 278—285.

Vairaktarakis, G. and M. Elhafsi (2000). "The use of flowlines to simplify routing complexity in two-stage flowshops". In: *IIE Transactions (Institute of Industrial Engineers)* 32.8, pp. 687–699.

Vallada, E. and R. Ruiz (2010). "Genetic algorithms with path relinking for the minimum tardiness permutation flowshop problem". In: *Omega* 38.1, pp. 57–67.

Wang, B. (1997). *Integrated product, process and enterprise design*. London. ISBN: 0412620200.

Wang, J., K. Tang, J. Lozano, and X. Yao (2015). "Estimation of Distribution Algorithm with Stochastic Local Search for Uncertain Capacitated Arc Routing Problems". In: *IEEE T. Evolut. Comput.* 20.c, pp. 1–1.

Wang, L., W. Shen, and Q. Hao (2006). "An overview of distributed process planning and its integration with scheduling". In: *International Journal of Computer Applications in Technology* 26.1/2, p. 3.

Yu, V. F. and S.-Y. Lin (2015). "A simulated annealing heuristic for the open location-routing problem". In: *Computers & Operations Research* 62, pp. 184–196.

Zhang, X. and S. Van De Velde (2012). "Approximation algorithms for the parallel flow shop problem". In: *European Journal of Operational Research* 216.3, pp. 544–552.

Zhou, Q. and X. Cui (2008). "Research on multiobjective flow shop scheduling with stochastic processing times and machine breakdowns". In: *IEEE International Conference on Service Operations and Logistics, and Informatics*. Vol. 2. IEEE, pp. 1718–1724.

B.7 ARPO: An iterated local search algorithm for rich portfolio optimization

Renatas Kizys[1], Angel A. Juan[2], Bartosz Sawik[3], Laura Calvet[2]

1. Subject Group of Economics and Finance, Portsmouth Business School, University of Portsmouth, UK
e-mail: renatas.kizys@port.ac.uk
2. Computer Science Department, IN3-Open University of Catalonia, Castelldefels, Spain
e-mail: {ajuanp, lcalvetl}@uoc.edu
3. Department of Applied Computer Science, AGH University of Science and Technology, Poland
e-mail: b_sawik@cal.berkeley.edu

Abstract

This research develops an original algorithm for rich portfolio optimization (ARPO), considering more realistic constraints than those usually analysed in the literature. Using a matheuristic framework that combines an iterated local search metaheuristic with quadratic programming, ARPO efficiently deals with complex variants of the mean-variance portfolio optimization problem, including the well-known cardinality and quantity constraints. ARPO proceeds in two steps. First, a feasible initial solution is constructed by allocating portfolio weights according to the individual return rate. Secondly, an iterated local search framework, which makes use of quadratic programming, gradually improves the initial solution throughout an iterative combination of a perturbation stage and a local search stage. According to the experimental results obtained, ARPO is very competitive when compared against existing state-of-the-art approaches, both in terms of the quality of the best solution generated as well as in terms of the computational times required to obtain it. Furthermore, we also show that our algorithm can be used to solve variants of the portfolio optimization problem, in which inputs (individual asset returns, variances and covariances) feature a random component. Notably, the results are similar to the benchmark constrained efficient frontier with deterministic inputs, if variances and covariances of individual asset returns comprise a random component. Finally, a sensitivity analysis has been carried out to test the stability of our algorithm against small variations in the input data.

Keywords: constrained portfolio optimization, metaheuristics, efficiency indices, financial assets, iterated local search, biased randomization.

1. Introduction

Since Markowitz (Markowitz, 1952), mean-variance optimization has become the workhorse model for portfolio selection. As discussed in Kolm et al. (2014), Markowitz' s theory of portfolio selection has had a major impact on academic research and on the industry of financial services[2]. An important assumption underlying this model is that investors are concerned about both the expected returns from their investment and the risk from that investment, where risk is defined as the variance of future returns. By optimally allocating appropriate weights to imperfectly correlated risky assets, investors can reduce the variance of future portfolio returns and thus diversify their investment.

Because of the non-negativity constraint on the level of investment in each asset, a closed analytical solution is generally not feasible (Maringer, 2005), and optimization methods need to be used to determine optimal portfolio weights. Since the mean-variance portfolio formulation, an enormous amount of papers have been published extending or modifying the basic model in three directions (Sawik, 2013a). The first path goes to simplification of the amount or the type of input data (Bertsimas and Pachamanova, 2008). The second path focuses on the introduction of an alternative measure of risk (Angelelli et al., 2007). The third path involves incorporation of the additional criteria and constraints (Perez et al., 2007). Traditional optimization methods include, among others, linear programming (Sharpe, 1967; Sharpe, 1971; Speranza, 1993; Sawik,

[2]Other recent studies on the theory of portfolio selection include Levy and Ritov (2011) and Leippold et al. (2011), among many others. Mean-variance portfolio optimization has also provided practical solutions to defined contribution pension schemes (Vigna, 2014), hedging mechanisms (Fujii and Takahashi, 2014), as well as to long-term fixed-income investments (Zumbach, 2013)

2012b; Mansini et al., 2014; Pae and Sabbaghi, 2014; Bruni et al., 2015),quadratic programming (Wolfe, 1959; Pachamanova and Fabozzi, 2010; Sawik, 2012a) and stochastic programming (Pınar, 2007). However, traditional optimization methods (notably, linear and quadratic programming) are plagued by a number of caveats. A notable characteristic of these methods is that they work only for problems that rely upon strict theoretical rules, albeit with a limited practical appeal (Kolm et al., 2014). Unfortunately, the complexity of the mean-variance model for portfolio optimization increases dramatically upon adding additional constraints that are needed to provide a rich representation of the investor's portfolio choice, which reduces the efficiency of exact methods in other than small-size instances. More recently, metaheuristic approaches have been proposed in the literature as a way to surpass these limitations (Maringer, 2005; Ólafsson, 2006; Jourdan et al., 2009; Blum et al., 2011; Boussaïd et al., 2013).

Accordingly, this paper focuses on a single-period version of the so-called constrained mean-variance portfolio optimization problem. We will assume in this work that all assets are risky assets (stocks). However, the solving approach introduced here could be also used for similar optimization problems regarding commodities, futures, options and swaps. Moreover, as in Tobin (1958) and Tobin (1965), our approach can be extended to include risk-free assets. The decision problem involves minimizing the portfolio variance for a given required rate of return. Portfolio weights add up to one and are constrained to take on non-negative values only. The latter assumption rules out short sales and thus places a constraint on excessive risk taking of investors. Under the above assumptions, the unconstrained efficient frontier (UEF) can be determined that gives, for each user-specified expected return, the minimum associated risk of investment. However, more realistic portfolio selection problems may involve additional constraints. First, justified on the grounds of the investor's preference and/or taste, the *pre-assignments* force some specific assets to be included in the portfolio. Second, the *quantity* constraint keeps the quantity of each selected asset within user-specified *floor* and *ceiling* values. The ceiling rules out excessive exposure to a specific asset. The floor is introduced in order to rule out the possibility of tiny (and therefore disproportionately costly) fractions of assets to be included in the portfolio. Third, the *cardinality* constraint, which imposes a floor and a ceiling on the number of assets included in the portfolio, accounts for the fact that diversification benefits decrease when the portfolio features a huge number of assets. When the above additional constraints are incorporated in the portfolio optimization problem, the constrained efficient frontier (CEF) can be obtained. In the presence of these rich constraints, the problem becomes NP-hard (Bienstock, 1996) and, thus, exact optimization methods quickly lose their efficiency as the number of considered assets grows. The cardinality constraint also implies that the mean-variance frontier can become discontinuous for certain values of expected return (Chang et al., 2000). In summary, as pointed out by Jobst et al. (2001), the cardinality and quantity constraints make large-size instances of the problem to be computationally intractable using traditional optimization approaches. Because our research involves the CEF, we devise a matheuristic algorithm for rich portfolio optimization (ARPO) that is based on the combination of an iterated local search (ILS) metaheuristic (Lourenço et al., 2010), quadratic programming, and biased randomization strategies (Juan et al., 2010; Juan et al., 2011b). The contribution of this study is fourfold. First, we show that using a carefully devised matheuristic solver can significantly reduce the minimum running time necessary to obtain near-optimal solutions. Indeed, as it will be discussed later, the computing-time performance of ARPO is significantly better than those of the solvers proposed by Schaerf (2002), Moral-Escudero et al. (2006), and Di Gaspero et al. (2011). Second, in terms of the minimum average percentage loss, the CEF determined by ARPO is approximately as close (or even closer in most cases) to the unconstrained efficient frontier as in the aforementioned studies. Third, the adoption of an initial solution that is based on a well-chosen criterion makes our solver more flexible when solving tight instances. The rate of return on an individual asset is used as a main criterion to construct the initial solution. It is worth noting that this criterion provides the best possible solution in terms of guarantying feasibility of the required portfolio return –i.e., if this initial solution is not feasible, then the problem has no feasible solution. By contrast, in most of the existing approaches, an initial solution is randomly drawn and, hence, the feasibility of these initial solutions cannot be guaranteed. Fourth and foremost, our paper relaxes the widely held assumption that inputs (individual asset returns, variances and covariances) are accurately measured and deterministic. In this regard, we show that ARPO can be used to solve portfolio optimization problems, in which a number of scenarios comprising uncertain returns, variances and covariances are studied and compared with the CEF-ARPO with accurate inputs. As expected, the CEF-ARPO with uncertain individual asset returns manifests

in a higher portfolio variance value for a given required rate of return. By contrast, no added uncertainty to individual asset returns translates into no material change to the CEF-ARPO with accurate inputs when variances and covariances between individual asset returns are subjected to a random disturbance. Finally, We further carry out a stability analysis – that involves small variations in cardinality and quantity constraints – shows no material deviations from the benchmark CEF in terms of the portfolio variance and computational time.

The remainder of this article is organized as follows. Section 2 provides a review of the literature related to the theme, also discussing the limitations of traditional methods and the need for new approaches based on metaheuristics. Section 3 gives a formal description of the optimization problem being considered, while Section 4 provides an overview of the ARPO algorithm, including its pseudo-code for quick implementation. Section 5 introduces the numerical experiments performed to test the algorithm performance, while the results are discussed in Section 6. Finally, Section 7 highlights the main contributions of this work and outlines plans for future research.

2. Need for new metaheuristic-based approaches

An updated literature review on the portfolio optimization problem can be found in Kolm et al. (2014). For this reason, our review focuses on analyzing the limitations of traditional approaches and discussing about the need of new algorithms that consider richer constraints and large-scale instances.

Traditional optimization methods feature a number of theoretical and practical limitations. A notable characteristic of these methods is that they work only for problems that typically rely upon strict deterministic rules. First, they can produce wrong solutions when the portfolio selection problem under consideration has one or more local maxima in addition to a global maximum (Maringer, 2005). Second, the optimal solution may be disguised by the presence of estimation errors in expected returns and variances that are used as inputs (Michaud, 1989; Kolm et al., 2014). Third, the optimal solution is notoriously unstable, as small changes in inputs can cause large changes in the optimal portfolio weights (Kallberg and Ziemba, 1984). Fourth, in terms of out-of-sample performance, traditional optimization methods are sometimes no better that the naïve portfolio, wherein all assets are allocated the same weight (DeMiguel et al., 2009). Along similar lines, Jorion (1985) argues that traditional optimization models can generate accurate in-sample forecasts, but their out-of-sample performance is generally weak. Fifth, a more realistic portfolio selection problem is typically confined to a subset of assets, which are selected by imposing additional constraints. A number of restrictions are specified in an investment management agreement between a client and a portfolio manager (Kolm et al., 2014). The client may impose a limit on the number of assets in the portfolio (*cardinality constraints*) (Maringer and Kellerer, 2003). Further, the client may also ask the manager to invest or not to invest in certain industries or companies. Therefore, the so-called *pre-assignment constraints* may be used that pre-assign certain industries or assets in the portfolio. The client may further impose a discretionary limit on exposure to certain industries or assets that aims at keeping the quantity of each asset within a given range (*quantity constraints*). Another battery of constraints is dictated by the presence of *transaction costs* (Kolm et al., 2014; Mansini et al., 2014). If one or more of these constraints are introduced in portfolio optimization, indeterminacies in the efficient portfolio frontier may arise further limiting the usefulness of traditional optimization methods.

In addition to the above theoretical limitations, traditional optimization methods have received a weak support in practice. Whilst diversification is a dominant portfolio selection strategy that contributes to lowering the risk of portfolio investment, in practice investors invest in fewer assets due to a variety of reasons. First, the administration of large portfolios can be cumbersome (Di Gaspero et al., 2011) or involve transaction costs in the form of bank and broker fees (Baule, 2010). Second, there is evidence that investors' desire to diversify is limited (Blume and Friend, 1975; Guiso et al., 1996; Jansen and Dijk, 2002). Third, Maringer (2005) and Pachamanova and Fabozzi (2010) argue that diversification can be achieved by investing in a small, yet well-chosen sub-set of assets. Similarly, Evans and Archer (1968) assert that, in most practical situations, investing in a portfolio comprising between 10 and 20 individual stocks can reasonably reduce the risk of investment. By the same token, Sharpe (1967) suggests including between 15 and 20 assets to diversify away most diversifiable risk. Lloyd et al. (1981) find that that, quite often, the optimal diversification should not use more than 27 assets in real-life investment. Moreover, a smaller number of assets imply less number of parameters to be estimated by using sample

information and, therefore, less room for estimation error (DeMiguel et al., 2009). Michaud (1989) provides a detailed account of advantages and disadvantages of traditional optimization methods. Specifically, he argues that these methods do not appeal to practitioners, since they do not make investment sense and do not have investment value. Such financial irrelevance is exacerbated by the fact that traditional optimization methods tend to over-weight assets with large returns and small variances and under-weight assets with small returns and large variances (Nawrocki, 2000). However, Kolm et al. (2014) maintain that the presence of theoretical and practical limitations does not invalidate Markowitz's theory of portfolio selection. Rather, traditional optimization methods need to be modified and new methods need to be developed, so that the existing gap between theory and practice can be bridged.

In order to address some of the theoretical and practical limitations of more classical approaches, metaheuristic methods to portfolio optimization have emerged in the literature (Maringer, 2005; Ólafsson, 2006; Jourdan et al., 2009; Blum et al., 2011; Boussaïd et al., 2013). Metaheuristics emerged from simple heuristics that were proposed to address some of the weaknesses inherited by traditional optimization methods. A heuristic is a common-sense and experience-based solution search method. Sharpe (1967) proposes a simple heuristic based on the beta as a risk measure to solve the portfolio selection problem. Elton et al. (1976) builds upon the expected return-variance ratio to develop a heuristic that determines an optimal portfolio location on the efficient frontier. More specifically, they construct a decision rule that determines whether the asset will enter the portfolio and calculates its weight in the portfolio. A simplification to the decision rule in Elton et al. (1976) is made by Nawrocki (1983). In particular, he derives a heuristic that is based on the expected return-semi-variance ratio that assumes the average correlation between securities to be zero. The use of the semi-variance as a measure of risk owes to the notion that investors are more concerned about downside risk than upside risk. Nawrocki (1983) demonstrates that the return-risk heuristic that uses the downside risk measure provides better investment performance than traditional optimization methods for long-term investments, albeit not for short-term ones. A comparison between the return-risk heuristic and two traditional optimization methods is performed by Nawrocki (2000). He finds that the return-risk heuristic produces the highest return along with the highest standard deviation and the lowest semi-deviation among the three portfolios. However, heuristics always return a local optimum, which may or may not be global optimum. The use of these simple return-risk heuristics for the constrained portfolio optimization problem has been discontinued in the last two decades. In fact, an increasing number of studies have tended to focus on metaheuristics. Metaheuristic search methods are less restrictive than traditional optimization methods and thus can be tailored to solve a particular optimization problem that features a number of constraints. Metaheuristics have been employed in several studies on portfolio optimization, such as Chang et al. (2000), Schaerf (2002), Crama and Schyns (2003), Derigs and Nickel (2003), Armañanzas and Lozano (2005), Moral-Escudero et al. (2006), Fernández and Gómez (2007), and Di Gaspero et al. (2011) to name just few. However, most of these studies feature the use of random initialization to the portfolio selection problem or simply downplay their strategies of portfolio initialization. Di Gaspero et al. (2011) are maybe an exception to this rule. In addition to the randomly generated initial solution, these authors propose two simple heuristics. First, they construct the portfolio that produces the maximum possible return, independently of the risk. Second, they use the final solution of the previously computed point on the efficient frontier. The amount of processor time it takes for the local search algorithm to reach the optimal solution on the efficient frontier may be shorter if a sensible simple heuristic is used to construct an initial solution.

3. Problem definition

In this section, the constrained mean-variance portfolio optimization problem is described. Section 3.1 provides an overview of the parameters, variables and constraints used in the problem. Section 3.2 contains the mathematical model, which is based on the one provided in Di Gaspero et al. (2011). For a comprehensive overview of the formulations of the portfolio selection problem, see Di Tollo and Roli (2008). Finally, Section 3.3 summarizes the algorithm's inputs and outputs.

3.1 Problem parameters, variables, and constraints

A set of n assets is given by the market, $A = \{a_1, a_2, \ldots, a_n\}$, where: *(a)* $\forall i \in \{1, 2, \ldots, n\}$, a_i has a known expected return, $r_i \geq 0$; *(b)* $\forall i, j \in \{1, 2, \ldots, n\}$, the pair (a_i, a_j) has a known expected risk index, $\sigma_{ij} = \sigma_{ji} \geq 0$; *(c)* a user-provided value, $R > 0$, represents the minimum expected return from the investment (expected return constraint); *(d)* a portfolio is a vector $X = (x_1, x_2, \ldots, x_n)$ such that each x_i represents the fraction of the total wealth invested in asset a_i, i.e., $0 \leq x_i \leq 1$ and $\sum_{i=1}^{n} x_i = 1$; *(e)* $\forall i \in \{1, 2, \ldots, n\}$, $z_i = 1$ if $x_i > 0$ (i.e., a_i in portfolio) and $z_i = 0$ otherwise; *(f)* the number of assets in the portfolio, $\sum_{i=1}^{n} z_i$, is bounded by user-defined values, k_{min} and k_{max} (cardinality constraints); *(g)* the user can pre-select certain assets to be included in the portfolio, i.e.: $\forall i \in \{1, 2, \ldots, n\}$, $p_i = 1$ if a_i is pre-assigned (i.e., $x_i > 0$) and $p_i = 0$ otherwise (pre-assignment constraints); and *(h)* for each asset a_i, its associated quantity in the portfolio, x_i, is bounded by user-defined values, ε_i and δ_i (quantity constraints). The typical objective of this problem is to select the optimal combination of fractions of each asset, x_i, so that the overall variance (risk) is minimized while satisfying all the aforementioned constraints.

3.2 The mathematical model

The mathematical model comprises an objective function and a set of constraints:

$$\min f(x) = \sum_{i=1}^{n} \sum_{j=1}^{n} \sigma_{ij} x_i x_j \tag{B.1}$$

subject to:

$$\sum_{i=1}^{n} r_i x_i \geq R \tag{B.2}$$

$$\sum_{i=1}^{n} x_i = 1 \tag{B.3}$$

$$0 \leq x_i \leq 1, \forall i \in \{1, 2, \ldots, n\} \tag{B.4}$$

$$k_{min} \leq \sum_{i=1}^{n} z_i \leq k_{max} \tag{B.5}$$

$$\varepsilon_i z_i \leq x_i \leq \delta_i z_i, \forall i \in \{1, 2, ..., n\} \tag{B.6}$$

$$0 \leq \varepsilon_i \leq \delta_i \leq 1, \forall i \in \{1, 2, ..., n\} \tag{B.7}$$

$$p_i \leq z_i, \forall i \in \{1, 2, ..., n\} \tag{B.8}$$

$$z_i \leq M x_i, \forall i \in \{1, 2, ..., n\} \tag{B.9}$$

$$z_i \in \{0, 1\}, \forall i \in \{1, 2, ..., n\} \tag{B.10}$$

Equation (1) describes the investor's objective function. The investor's objective is to minimize the portfolio variance. Equations (1) – (4) outline the basic ("unconstrained") optimization problem and determine the UEF. Specifically, Equation (2) provides the lower bound for the investor's required return. Equation (3) ensures that portfolio weights add up to unity. The purpose of Equation (4) is to regulate leveraged positions. This equation is justified on the grounds of the existing short selling regulations in a wide range of countries (Jain et al., 2013). A number of realistic situations often require additional constraints that can be summarized by means of Equations (5) – (10). By solving the constrained optimization problem given by Equations (1) – (10) the so-called constrained efficient frontier (CEF) is obtained. Equation (5) formulates cardinality constraints. Equation (6) defines quantity constraints. The quantity of each asset a_i is confined to a given range. A minimum quantity of wealth invested in asset a_i is given by ε_i. A maximum

quantity of wealth invested in asset a_i is given by δ_i. Both parameters ε_i and δ_i range from 0 to 1 (Equation (7)). Note that Equation (4) is redundant when Equations (6) and (7) are considered. Equations (8) and (9) imply that certain assets are pre-assigned in the portfolio. In particular, given a vector of n binary decision variables Z (Equation (10)) (where z_i takes on value 1 if included in the portfolio and 0 otherwise), and a binary vector P of pre-assignments (in which p_i takes on value 1 if pre-assigned and 0 otherwise), whenever asset a_i is pre-assigned, it has to be included in the portfolio (Equation (8)). In Equation (9), M is a large positive value such that $Mx_i \geq 1$ for all $x_i \geq 0$. Thus, if the quantity in the portfolio of asset a_i, x_i, is equal to 0, it means that this asset is not included in the portfolio (i.e., $z_i = 0$). Only a few papers have considered the use of pre-assignment constraints. An example is Di Gaspero et al. (2011), who found that the CEF obtained by pre-assigning a high-yield asset strictly dominates the CEF obtained by pre-assigning a low-yield asset. In general, they find an inverse relation between the average percentage loss (with regard to the UEF) and the return value. However, they do not report computing times for the CEF with pre-assignment constraints, which makes impossible to complete a fair comparison with their results.

3.3 Algorithm inputs and outputs

According to the problem description, the output of the algorithm will be an assets-investment plan (solution), $X = (x_1, x_2, \ldots, x_n)$, satisfying all the aforementioned constraints and with the lowest possible risk, $f(X)$. Similarly, the inputs of the algorithm are the following ones: *(a)* for each asset $a_i \in A$, the following values: r_i, ε_i, and δ_i; *(b)* the matrix of co-variances: $\{\sigma_{ij}/i \leq j\}$; *(c)* the user-defined minimum expected return: $R > 0$; and *(d)* the boundaries on the number of assets to include: k_{min} and k_{max}. In order to perform a fair comparison with some previous works and existing benchmarks (Chang et al., 2000; Schaerf, 2002; Armañanzas and Lozano, 2005; Moral-Escudero et al., 2006; Fernández and Gómez, 2007), in this paper we will not consider pre-selected assets. However, due to its flexibility, the ARPO algorithm could be adapted without too much effort to deal with this constraint too.

4. The ARPO metaheuristic

The ARPO matheuristic combines three main components: *(a)* an ILS framework (Lourenço et al., 2010); *(b)* the use of a biased randomization process (Juan et al., 2010; Juan et al., 2011b) that guides the generation of new 'promising' solutions (perturbation stage); and *(c)* the use of a quadratic programming solver that, given a current portfolio, optimizes the levels of investment of each asset (local search). ILS is a conceptually simple yet powerful metaheuristic that has proven to be very efficient in solving complex combinatorial optimization problems. The underlying idea behind ILS is to narrow the search for candidate local optimal solutions returned by some embedded algorithm, typically a local search heuristic. Burke et al. (2010) show that ILS obtains the best average performance among a set of selected metaheuristic approaches in three classical combinatorial optimization problems: bin packing, permutation flow shop, and personnel scheduling. The authors also emphasize two main factors for its success: *(i)* an excellent balance between exploration and exploitation by "systematically combining a perturbation followed by local search"; and *(ii)* its relative simplicity and the reduced number of parameters required, factors that facilitate its quick implementation in practical applications.

Pseudo-code 1 shows the main procedure of the ARPO algorithm. Apart from the inputs defining the instance, also the maximum computing time allowed, *maxTime*, and an additional parameter, *beta*, are passed to the procedure –the use of this additional parameter will be discussed later.

The ARPO procedure starts by generating a 'dummy' initial solution (line 01). This initial solution is constructed by including the assets with the highest return levels so that it provides the highest possible expected return while satisfying all the remaining constraints. This way, if the expected return provided by this solution does not reach the minimum return threshold imposed by the investor, then the problem will be infeasible since no other solution will do it (lines 02-04). Notice, however, that it is also likely to obtain a high risk associated with this initial solution –hence the name dummy.

At this point, a quick local search (lines 06-07) is applied to this initial solution in order to improve it without losing its feasibility. This local search uses quadratic programming in order to

Algorithm 1 Main procedure of the ARPO algorithm (ILS framework).

```
    procedure ARPO(inputs, minReturn, maxTime, beta)
 1: initSol ← genInitSol(inputs)                    ▷ generate sol with highest possible return rate
 2: if {getReturn(initSol) < minReturn} then
 3:     return unfeasible                                                    ▷ unfeasible problem
 4: end if
 5: genFriendshipLists(inputs)                ▷ generate a sorted list of "friends" for each asset
 6: baseSol ← QPOptimize(initSol, minReturn)          ▷ optimize levels for each asset in portf.
 7: baseSol ← cleanSol(baseSol)                       ▷ delete from portf. assets with level = 0
 8: bestSol ← baseSol                                                    ▷ initialize bestSol
 9: elapsedTime ← 0
10: credit ← 0                                                ▷ used in the acceptance criterion
11: while {elapsedTime < maxTime} do                                      ▷ iterated local search
12:     newSol ← perturbateSol(baseSol, inputs, beta)          ▷ destruction-construction stages
13:     if {getMaxReturnAsset(newSol) < minReturn} then          ▷ fix solution if unfeasible
14:         newSol ← repairSol(newSol, inputs)
15:     end if
16:     if {newSol is in cache} then                                  ▷ already optimized levels
17:         newSol ← loadFromCache(newSol)              ▷ use optimized levels saved in cache
18:     else          ▷ apply a local search based on quadratic programming optimization
19:         newSol ← QPOptimize(newSol, minReturn)       ▷ optimize levels f.e. asset in portf.
20:         newSol ← cleanSol(newSol)                    ▷ delete from portf. assets with level = 0
21:         saveInCache(newSol)
22:     end if
23:     delta ← getRisk(newSol) - getRisk(baseSol)                ▷ newSol improves baseSol
24:     if {delta < 0} then
25:         credit ← -delta
26:         baseSol ← newSol
27:         if {getRisk(newSol) < getRisk(bestSol)} then          ▷ newSol improves bestSol
28:             bestSol ← newSol
29:         end if
30:     else{delta > 0 and delta ≤ credit}                            ▷ acceptance criterion
31:         credit ← 0
32:         baseSol ← newSol
33:     end if
34:     update elapsedTime
35: end while
36: return bestSol
    end procedure
```

optimize the investment level assigned to each asset in the current portfolio (solution). The improved solution will be considered both as the current 'base' solution and the 'best-so-far' solution (line 08). Now, the ARPO procedure resumes by starting an iterative improvement process (lines 11-35). As in most ILS frameworks, this process comprises three stages: *(a)* the *perturbation* stage (lines 12-15), which applies strong changes to the current base solution in order to increase exploration of the space of solutions; *(b)* the *local search* stage (lines 16-22), which tries to perform a quick improvement of the current base solution by applying some operators –in our case, it is based on the combined use of quadratic programming and a cache memory; and *(c)* the *acceptation* stage (lines 23-33), which in our case makes use of a credit-based system in order to allow accepting, under certain restrictive conditions, a new base solution even when it offers a slightly higher risk than the current base solution –this 'degradation' of the base solution is allowed in order to reduce the probabilities of getting trapped in a local minimum during the searching process.

As regards as the *perturbation* stage (Pseudo-code 2), this follows a destruction - reconstruction process. First, this process takes as an input the current base solution. Second, the current base solution is partially destroyed according to some random criterion –in our case, a randomly selected number of assets are deleted from the portfolio– (lines 01-09). Third, the destroyed solution is re-constructed (completed) by adding new assets to the portfolio (lines 10-19).

Algorithm 2 Perturbation procedure to generate new 'promising' solutions.

```
    procedure perturbateSol(baseSol, inputs, beta)
 1: newSol ← copySol(baseSol)
                    ▷ 1. Remove a random number of randomly selected assets (destruction stage)
 2: nAssetsInSol ← getNAssetsInSol(newSol)
 3: if {nAssetsInSol > 1} then
 4:     nAssetsToRemove ← genRandomNumber(1, nAssetsInSol - 1)
 5:     for {i = 1 to nAssetsToRemove} do
 6:         asset ← selectRandomAsset(newSol)
 7:         newSol ← removeAsset(asset, newSol)
 8:     end for
 9: end if
                        ▷ 2. Randomly select one asset in current portf. to add several of its "friends"
10: asset ← getRandomAsset(newSol)
          ▷ 3. Use biased rand. to add friendly assets until reaching kMax (re-construction stage)
11: while {size(newSol) < getKMax(inputs)} do
12:     listOfFriendlyAssets ← getFriendlyList(asset)          ▷ Sorted list of friendly assets
13:     do          ▷ Randomly select a position using a Geometric(beta) prob. distribution
14:         position ← biasedRandom(size(listOfFriendlyAssets), beta)
15:         newAsset ← getAsset(listOfFriendlyAssets, position)
16:     while {newAsset in newSol}          ▷ Repeat until newAsset not in current portf.
17:         newSol ← addAsset(newAsset, newSol)
18:         asset ← newAsset
19: end while
20: return newSol
    end procedure
```

During this re-construction process, the selection of each new asset added to the portfolio is done following a 'friendship' criterion, i.e.: although the selection of the new asset is random, this new asset will be most likely selected among those assets that are highly compatible –i.e., showing a low covariance value– with the last asset added to the portfolio. This special behavior is attained throughout the use of a biased randomization selection process (line 14), which makes use of a geometric distribution of parameter beta ($0 < beta < 1$). More details on biased randomization processes can be found in Juan et al. (2010) and Juan et al. (2011b).

Finally, there might be times in which the newly generated solution does not fulfil the minimum return requirement. In those cases, a 'repair' stage is used to swap a randomly selected asset in the current portfolio (lines 01 – 04) by a high-return asset not currently in the portfolio (lines 05 – 10) (Pseudo-code 3).

Algorithm 3 Repair procedure to make newly generated solutions feasible.

```
    procedure perturbateSol(baseSol, inputs, beta) procedure repairSolution(sol, inputs)
 1: unusedAssets ← getAssetsNotInSol(sol, inputs)          ▷ Consider assets not in portf.
 2: unusedAssets ← shuffle(unusedAssets)          ▷ Random sorting of the unused assets list
 3: assetA ← getRandomAsset(sol)          ▷ Select a random assetA in current portf.
 4: sol ← deleteAsset(assetA, sol)          ▷ Delete assetA from current portf.
 5: for {each asset assetB in unusedAssets} do          ▷ Search unused assetB with high return
 6:     if {getReturn(assetB) ≥ minReturn} then
 7:         sol ← addAsset(assetB, sol)          ▷ Add assetB to current port.
 8:         return sol
 9:     end if
10: end for
11: end procedure
```

5. Numerical experiments

The ARPO algorithm has been implemented as a Java application. Being an interpreted language, Java-based programs do not execute as fast as other compiled programs, such as those developed in C or C++. Nevertheless, Java permits a rapid, platform-independent, development of object-oriented prototypes that can be used to test the potential of an algorithm. Also, using Java allowed to integrate our code with ojAlgo (http://ojalgo.org), an open-source quadratic programming solver developed in Java. A standard personal computer, Intel Core i5 CPU at 3.2 GHz and 4 GB RAM with Linux Ubuntu, was used to perform all tests.

In this research, we experiment with two sets of stock market data already used in previous studies. The first set of data was retrieved from the repository ORlib, and it can be downloaded from the following website: http://people.brunel.ac.uk/~mastjjb/jeb/orlib/portinfo.html. These instances were proposed by Chang et al. (2000) and were studied by Schaerf (2002), Armañanzas and Lozano (2005), Moral-Escudero et al. (2006), Fernández and Gómez (2007), and Di Gaspero et al. (2011). The data set comprises constituents of five stock market indices, Hang Seng (Hong Kong), DAX 100 (Germany), FTSE 100 (United Kingdom), S&P 100 (United States) and NIKKEI 225 (Japan). These indices were extracted from DataStream and are measured at weekly frequency spanning the period from March 1992 to September 1997.

Following Di Gaspero et al. (2011) we divided the portfolio frontier into 100 equidistant points on the vertical axis that represents the user-defined rate of expected portfolio return. Although the algorithm has been designed for the constrained case, it is initially tested on the unconstrained mean-variance optimization problem. The test results show that our solver is able to return solutions that are overlapping with the unconstrained efficient frontier (UEF) published at the OR Library, which contributes to validate the effectiveness of our approach.

Next, we execute the algorithm on a constrained mean-variance frontier (the algorithm is executed 30 times and both the best and average results are recorded). The maximum time of execution for each instance is 20 seconds. The benchmark constraints are those imposed by the previous authors. Essentially, the constraints involve the following conditions: $\varepsilon_i = 0.01$, $\delta_i = 1$, $k_{min} = 1$, $k_{max} = 10$, $\forall i \in \{1, 2, \ldots, n\}$. As in the aforementioned studies, pre-assignment constraints are not considered in these experiments, i.e., $p_i = 0$, $\forall i \in \{1, 2, \ldots, n\}$. Notice that, despite other authors claim that their approaches can solve the constrained problem with all the aforementioned constraints, this fact is not clearly showed neither in the description of their methods nor in the benchmarks they solve, since the parameter values they use in their benchmarks do not seem to impose a real challenge for their algorithms in terms of tight constraints.

6. Discussion of results

CEF-ARPO with certain inputs

Table B.1 shows the values of average percentage loss (APL) and associated computational times. Notice that, in terms of the minimum APL, our ARPO algorithm outperforms on Instances 2 – 5 the hybrid solvers proposed by Di Gaspero et al. (2011), which comprise combinations of first descent and steepest descent with quadratic programming (FD+QP and SD+QP, respectively). With regard to the first instance, our APL is greater, but this result may emanate from rounding errors. In terms of computational time, ARPO shows a superior performance relative to that of the solver's SD+QP and is comparable or better than the solver's FP+QP performance. We next contrast our results with the results reported by Schaerf (2002) and Moral-Escudero et al. (2006). Although the minimum APL provided by ARPO is slightly superior to the hybrid solver combining a genetic algorithm (GA) and quadratic programming (QP) in Moral-Escudero et al. (2006), on the remaining instances the minimum APL accomplished by ARPO is lower. Furthermore, our computational times are considerably lower than those reported by the tabu search (TS) in Schaerf (2002), and by GA+QP in Moral-Escudero et al. (2006).

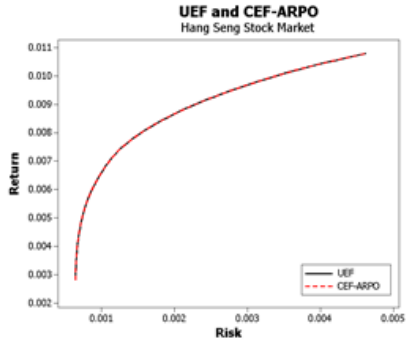
The UEF (as provided in the ORlib) and CEF (as provided by ARPO) for the five stock market indices are compared in Panels A – E of Figure B.1.

Panel A of Figure B.1 depicts the CEF for the Hang Seng (Hong Kong) stock market. A visual inspection suggests that for the Hang Seng stock index the CEF is hardly distinguishable from the UEF. However, as the rate of expected return increases, along with increasing risk of investment, the CEF tends to diverge relatively less from the UEF. In particular, at the higher end of the CEF

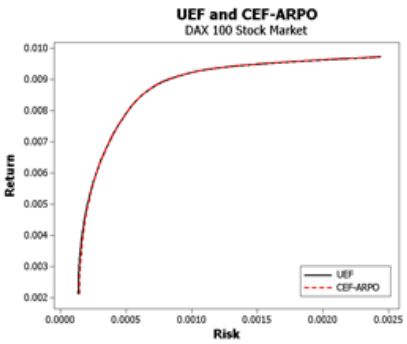
TABLE B.1: Summary of Results

Instance	Di Gaspero et al. (2011)				Moral-Escudero et al. (2006)		Schaerf (2002)		ARPO	
	FD + QP		SD + QP		GA + QP		TS		LS + QP	
	APL	T(s)	APL	T(s)	APL	T(s)	APL	T(s)	APL	T(Std) (s)
HS	0.00366	1.5	0.00321	3.1	0.00321	415.1	0.00409	251	0.00399	2.0 (0.87)
DAX 100	2.66104	9.6	2.53139	14.1	2.53180	552.7	2.53617	531	2.45403	1.0 (1.0)
FTSE 100	2.00146	10.1	1.92146	16.1	1.92150	886.3	1.92597	583	1.88340	13.7 (9.28)
S&P 100	4.77157	11.2	4.69371	18.8	4.69507	1163.7	4.69507	713	4.65095	15.5 (8.42)
NIKKEI 225	0.24176	25.3	0.20219	45.9	0.20198	1465.8	0.20198	1603	0.20189	5.0 (17.20)

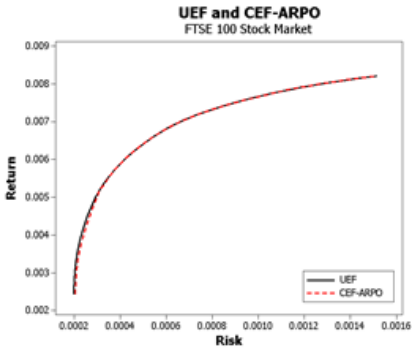
Table 1 summarizes the results of ARPO for the CEF, including the minimum average percentage loss (APL) and computational time in seconds (T(s)). APL is calculated as in Di Gaspero et al. (2011). We also report the standard deviation of computational time for ARPO. The performance indicators of ARPO are also compared against its competitors, including Di Gaspero et al. (2011), Moral-Escudero et al. (2006), and Schaerf (2002).



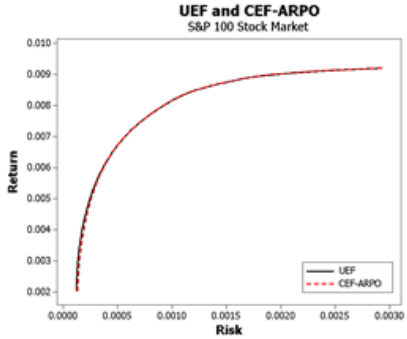
Panel A – Hang Seng Stock Market (Hong Kong)



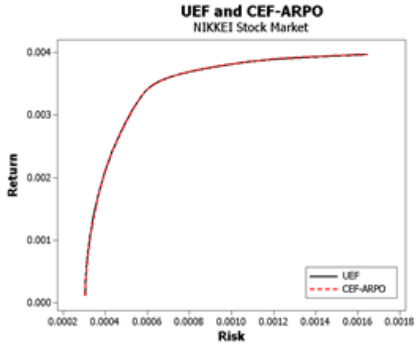
Panel B – DAX 100 Stock Market (Germany)



Panel C – FTSE 100 Stock Market (UK)



Panel D – S&P 100 Stock Market (US)



Panel E – NIKKEI Stock Market (Japan)

FIGURE B.1: UEF and CEF-ARPO

Notes: Figure B.1 illustrates the UEF and the CEF-ARPO for the five instances (Panel A – Hang Seng, Panel B – DAX 100, Panel C – FTSE 100, Panel D – S&P 100 and Panel E – NIKKEI 225). The UEF is represented with a black solid line, whereas the CEF-ARPO is represented with a dash red line. Portfolio variance is depicted on the horizontal axis, whereas the required rate of return is depicted on the vertical axis. The minimum average percentage loss is 0.00399% for the Hang Seng portfolio, 2.45403% for the DAX 100 portfolio, 1.88340% for the FTSE100 portfolio, 4.65095% for the S&P 100 portfolio and 0.20189% for the NIKKEI 225 portfolio.

TABLE B.2: Hang Seng Stock Market (Hong Kong).

Curve Position	Required Return	UEF-Variance	ARPO-Variance	APL	Time (s)
20	0.002861137	0.0006424068	0.0006424114	0.0007160572	4.212
40	0.002941981	0.0006428092	0.0006429074	0.0152766949	2.74
60	0.003022827	0.0006434196	0.0006437456	0.0506667811	1.978
80	0.003103671	0.0006442382	0.0006443922	0.0239042019	3.759
100	0.003184516	0.0006452648	0.0006454721	0.0321263456	1.226
120	0.003265361	0.0006464996	0.0006467783	0.0431090754	1.917
140	0.003346206	0.0006479424	0.0006483109	0.0568723393	2.012
160	0.003427051	0.0006495933	0.0006499731	0.0584673518	2.686
180	0.003507896	0.0006514524	0.0006516646	0.0325733699	1.491
200	0.003588740	0.0006535208	0.0006536148	0.0143836279	4.464
1820	0.010137479	0.0035773525	0.0035773526	0.0000027954	0.004
1840	0.010218315	0.0036907539	0.0036907539	0.0000000000	0.009
1860	0.010299151	0.0038090873	0.0038090873	0.0000000000	0.001
1880	0.010379986	0.0039323522	0.0039323522	0.0000000000	0.009
1900	0.010460822	0.0040605480	0.0040605480	0.0000000000	0.001
1920	0.010541657	0.0041936758	0.0041936758	0.0000000000	0.003
1940	0.010622493	0.0043317350	0.0043317350	0.0000000000	0.010
1960	0.010703329	0.0044747255	0.0044747255	0.0000000000	0.001
1980	0.010784164	0.0046226475	0.0046226476	0.0000021633	0.003
2000	0.010865000	0.0047755010	0.0047755010	0.0000000000	0

Notes: Table B.2 summarizes the UEF and the CEF-ARPO for the Hang Seng portfolio for 10 initial points and 10 last values of returns. In column 2, values of the required rate of return are provided. In column 3, values of the UEF solution (portfolio variance) are provided. In column 4, values of the CEF-ARPO solution (portfolio variance) are provided. In column 5, values of the minimum average percentage loss are reported. In column 6, computational times are reported.

TABLE B.3: DAX 100 Stock Market (Germany).

Curve Position	Required Return	UEF-Variance	ARPO-Variance	APL	Time (s)
20	0.002175078	0.0001368925	0.0001481318	0.0821031101	1.5320
40	0.002252039	0.0001370119	0.0001483250	0.0825702001	0.2180
60	0.002329001	0.0001372175	0.0001484472	0.0818386868	0.3910
80	0.002405963	0.0001375210	0.0001485474	0.0801797544	0.8330
100	0.002482925	0.0001379123	0.0001486181	0.0776275938	1.8510
120	0.002559887	0.0001383842	0.0001487104	0.0746197904	0.8630
140	0.002636849	0.0001389376	0.0001490367	0.0726880269	0.5330
160	0.00271381	0.0001395866	0.0001495985	0.0717253662	0.5620
180	0.002790771	0.0001403353	0.0001503958	0.0716890191	0.9450
200	0.002867732	0.0001411837	0.0001514287	0.0725650341	1.2600
1820	0.009101412	0.0008965075	0.0008965075	0.0000000000	0.0000
1840	0.009178374	0.0009614987	0.0009614987	0.0000000000	0.0000
1860	0.009255336	0.0010349696	0.0010351065	0.0001322744	0.0000
1880	0.009332288	0.0011354764	0.0011354764	0.0000000000	0.0000
1900	0.009409241	0.0012881113	0.0012881113	0.0000000000	0.0000
1920	0.009486192	0.0014930083	0.0014930083	0.0000000000	0.0000
1940	0.009563145	0.0017501725	0.0017501725	0.0000000000	0.0000
1960	0.009640096	0.0020595971	0.0020595971	0.0000000000	0.0000
1980	0.009717049	0.0024212903	0.0024212904	0.0000000413	0.0000
2000	0.009794000	0.0028352430	0.0028352430	0.0000000000	0.0000

Notes: Table B.3 summarizes the UEF and the CEF-ARPO for the DAX 100 portfolio for 10 initial points and 10 last values of returns. In column 2, values of the required rate of return are provided. In column 3, values of the UEF solution (portfolio variance) are provided. In column 4, values of the CEF-ARPO solution (portfolio variance) are provided. In column 5, values of the minimum average percentage loss are reported. In column 6, computational times are reported.

Table B.4: FTSE 100 Stock Market (United Kingdom).

Curve Position	Required Return	UEF-Variance	ARPO-Variance	APL	Time (s)
20	0.002420865	0.0001985238	0.0002060320	0.037820	24.1290
40	0.002479328	0.0001986154	0.0002061982	0.038178	4.7360
60	0.002537792	0.0001987642	0.0002065801	0.039322	21.8640
80	0.002596256	0.0001989714	0.0002066068	0.038374	16.4550
100	0.00265472	0.0001992442	0.0002068282	0.038064	25.5930
120	0.002713184	0.0001995842	0.0002073013	0.038666	13.9090
140	0.002771647	0.0001999959	0.0002080261	0.040152	26.5100
160	0.002830111	0.0002004890	0.0002084997	0.039956	18.1160
180	0.002888575	0.0002010665	0.0002089129	0.039024	15.0670
200	0.002947039	0.0002017309	0.0002095566	0.038793	17.4660
1820	0.007682888	0.0010170776	0.0010170776	0.000000	1.3320
1840	0.007741359	0.0010578347	0.0010581058	0.000256	0.5650
1860	0.00779983	0.0011002537	0.0011003206	0.000061	11.5260
1880	0.007858284	0.0011455465	0.0011455466	0.000000	13.3580
1900	0.007916738	0.0011954685	0.0011954685	0.000000	3.9120
1920	0.007975191	0.0012500871	0.0012500871	0.000000	28.7840
1940	0.008033645	0.0013094021	0.0013094021	0.000000	27.6720
1960	0.008092098	0.0013734126	0.0013734126	0.000000	26.4330
1980	0.008150551	0.0014421206	0.0014423115	0.000132	19.7400
2000	0.008209000	0.0015166351	0.0015166351	0.000000	0.0000

Notes: Table B.4 summarizes the UEF and the CEF-ARPO for the FTSE 100 portfolio for 10 initial points and 10 last values of returns. In column 2, values of the required rate of return are provided. In column 3, values of the UEF solution (portfolio variance) are provided. In column 4, values of the CEF-ARPO solution (portfolio variance) are provided. In column 5, values of the minimum average percentage loss are reported. In column 6, computational times are reported.

Table B.5: S&P 100 Stock Market (United States).

Curve Position	Required Return	UEF-Variance	ARPO-Variance	APL	Time (s)
20	0.002005874	0.0001214699	0.000134226	0.105014	19.1900
40	0.002078497	0.0001216461	0.000134619	0.106648	26.0990
60	0.002151121	0.0001219398	0.000135389	0.110297	20.5970
80	0.002223742	0.0001223689	0.000136242	0.113373	17.2350
100	0.002296365	0.0001229290	0.000137073	0.115057	22.1810
120	0.002368987	0.0001236105	0.000138075	0.117013	23.3780
140	0.00244161	0.0001244126	0.000139442	0.120799	22.3860
160	0.002514232	0.0001253355	0.000140429	0.120421	19.7720
180	0.002586853	0.0001263852	0.000141311	0.118098	26.2040
200	0.002659475	0.0001275649	0.000142711	0.118735	16.2610
1820	0.008541599	0.0012695539	0.001269554	0.000000	0.6500
1840	0.008614195	0.0013438638	0.001343864	0.000000	0.3990
1860	0.008686789	0.0014260901	0.00142609	0.000000	2.2590
1880	0.008759385	0.0015162347	0.001516235	0.000000	0.1940
1900	0.008831981	0.0016142967	0.001614297	0.000000	14.5950
1920	0.008904579	0.0017205530	0.001720553	0.000000	0.1340
1940	0.008977215	0.0018772540	0.001877254	0.000000	19.9680
1960	0.009049852	0.0021147413	0.002114741	0.000000	22.3300
1980	0.009122459	0.0024387539	0.002438754	0.000000	0.0070
2000	0.009195000	0.0029387241	0.002938724	0.000000	0.0000

Notes: Table B.5 summarizes the UEF and the CEF-ARPO for the S&P 100 portfolio for 10 initial points and 10 last values of returns. In column 2, values of the required rate of return are provided. In column 3, values of the UEF solution (portfolio variance) are provided. In column 4, values of the CEF-ARPO solution (portfolio variance) are provided. In column 5, values of the minimum average percentage loss are reported. In column 6, computational times are reported.

TABLE B.6: NIKKEI Stock Market (Japan).

Curve Position	Required Return	UEF-Variance	ARPO-Variance	APL	Time (s)
20	0.0001078963	0.0003046821	0.0003048207	0.000455	0.0340
40	0.0001469218	0.0003048095	0.0003049299	0.000395	0.0820
60	0.0001859471	0.0003050116	0.0003051331	0.000398	0.0850
80	0.0002249721	0.0003052881	0.0003054303	0.000466	0.0660
100	0.0002639974	0.0003056390	0.0003058216	0.000597	0.1000
120	0.0003030223	0.0003060641	0.0003063069	0.000793	0.1930
140	0.0003420475	0.0003065635	0.0003068863	0.001053	0.1930
160	0.0003810730	0.0003071384	0.0003075597	0.001372	0.0540
180	0.0004200985	0.0003077930	0.0003083271	0.001735	0.1150
200	0.0004591229	0.0003085244	0.0003091886	0.002153	0.0400
1820	0.0036202364	0.0007178707	0.0007178707	0.000000	0.0320
1840	0.0036592244	0.0007585282	0.0007585282	0.000000	0.0210
1860	0.0036982126	0.0008065920	0.0008065920	0.000000	0.0520
1880	0.0037372032	0.0008621079	0.0008621079	0.000000	60.6030
1900	0.0037762020	0.0009261274	0.0009261274	0.000000	0.0050
1920	0.0038152008	0.0009990847	0.0009990847	0.000000	0.0050
1940	0.0038541986	0.0010809831	0.0010809831	0.000000	85.8630
1960	0.0038931362	0.0011966840	0.0011966841	0.000000	0.0010
1980	0.0039320680	0.0013855561	0.0013855562	0.000000	0.0010
2000	0.0039710000	0.0016485224	0.0016485224	0.000000	0.0000

Notes: Table B.6 summarizes the UEF and the CEF-ARPO for the NIKKEI 225 portfolio for 10 initial points and 10 last values of returns. In column 2, values of the required rate of return are provided. In column 3, values of the UEF solution (portfolio variance) are provided. In column 4, values of the CEF-ARPO solution (portfolio variance) are provided. In column 5, values of the minimum average percentage loss are reported. In column 6, computational times are reported.

that features rewarding but risky portfolios, the expected rate of return can be attained with fewer assets.

Panel B of Figure B.1 depicts the CEF for the DAX 100 (Germany) stock market. Visual inspection indicates that for the DAX 100 stock index the CEF diverges from the UEF at the lower end of expected return, more specifically, for $R < 0.006$. As the rate of expected return increases, the CEF becomes indistinguishable from the UEF. Notably, portfolios with an expected return at the lower end of the CEF tend to be riskier (i.e., with higher portfolio variance) than portfolios on the UEF.

Panel C of Figure B.1 depicts the CEF for the FTSE 100 (United Kingdom) stock market. It indicates that for the FTSE 100 stock index –similarly to the DAX 100 stock index– the CEF departs from the UEF at the lower end of expected return. As the rate of expected return increases, the CEF converges to the UEF. Noteworthy, portfolios featuring an expected return at the lower end of the CEF tend to be riskier (i.e., with higher portfolio variance) than portfolios on the UEF. At the higher end of the CEF that features rewarding but risky portfolios, the expected rate of return can be achieved with fewer assets.

Panel D of Figure B.1 depicts the CEF for the S&P 500 (United States) stock market. It indicates that for the S&P 500 stock index –as with the DAX 100 and FTSE 100 stock indices– the CEF departs from the UEF at the lower end of expected return. As the rate of expected return increases, the CEF becomes visually indistinguishable from the UEF. Portfolio investments with an expected return at the lower end of the CEF involve relatively more risk than portfolios with the same expected return located on the UEF.

Finally, Panel E of Figure B.1 depicts the CEF for the NIKKEI (United States) stock market. It indicates that for the NIKKEI stock index, the relation between the CEF and the UEF follows a pattern similar to the Hang Seng stock index. Specifically, although the CEF departs from the UEF at the lower end of expected return, the difference is visually very small. As the rate of expected return increases, the CEF gradually approaches the UEF. At the higher end of the CEF that includes portfolios with high expected return and high risk, the APL approaches to zero.

To evaluate differences between the UEF and the CEF-ARPO, we also provide the portfolio weights for Instance 3 (FTSE 100), where the required rate of return is 0.0041572635, which is an approximately central value within the overall of returns. This instance was executed twice with

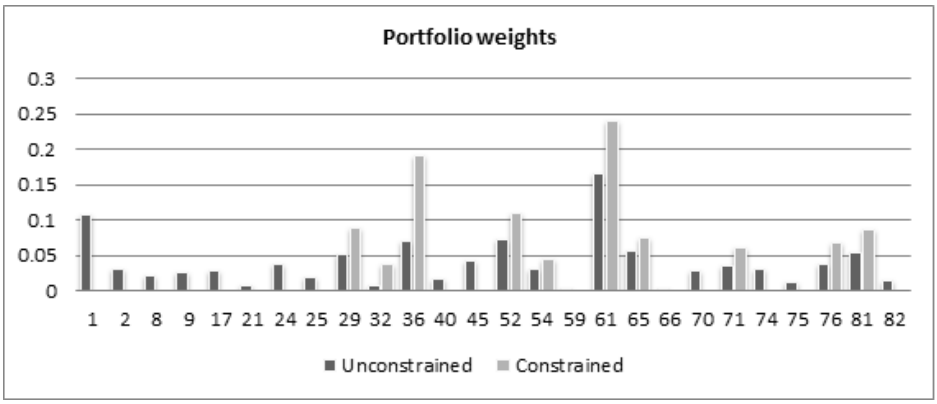


FIGURE B.2: Portfolio Weights

Notes: The horizontal axis indexes assets, whereas the vertical axis measures the weight of an asset.

the same seed, where the maximum time of execution was 20 seconds. The UEF considered all assets with weights ranging from 0 to 1 inclusively. The CEF was constrained to the minimum of 1 and the maximum of 10 assets, with portfolio weights ranging from 0.01 to 1. The minimum values of the portfolio variance were 2.3872556507357437E-4 (the UEF) and 2.5098945345432527E-4 (CEF-ARPO). The percentage loss is 5.137%. The UEF selected 26 assets (the remaining assets were allocated zero weight), whereas the CEF portfolio selected 10 assets, the upper bound of the cardinality constraint. The 10 assets are the subset of assets selected in the UEF portfolio, where assets indexed with 61, 36 and 52 carry the largest weight in the CEF portfolio.

CEF-ARPO with uncertain inputs

Key to our research is a widely held assumption –in line with most of the existing literature –that the returns vector and the variance and covariance matrix are well-know and deterministic; i.e., they are known with certainty. Whilst this assumption is analytically and computationally convenient, its practical appeal may be limited. In order to account for this practical restraint, and to test if our algorithm is robust to small changes in the returns vector and the variance-covariance matrix, we use the data set of Hang Seng stock market to perform computational experiments that involve some degree of uncertainty in the first and second moments. We consider three different scenarios. In Scenario 1 (S1), a random disturbance term distributed with a normal distribution with mean 0 and standard deviation 0.00015 is added to each return of the original instance. Under this scenario, an original return of 0.00350406 (which is the mean return of the instance) now takes on values within the interval of 0.00350406±2·0.00015=(0.00320406,0.00380406) with the probability of 95.45%, which can be deemed a realistic interval. In Scenario 2 (S2), a random disturbance term distributed with a normal distribution with mean 0 and standard deviation 0.000025 is added to each element of the covariance matrix. Under this scenario, the original covariance of 0.00113094 now takes on values within the interval of (0.00108094,0.00118094) with the probability of 95.45%. In Scenario 3 (S3), Scenarios 1 and 2 are jointly simulated. The results are compared against Scenario 0 (S0), which depicts the CEF–ARPO with $k_{min} = 1$, $k_{max} = 10$, and portfolio weights confined between 0.01 and 1.

Table B.7 reports the portfolio variance and the APL loss relative to S0 of the three scenarios, for 10 initial and 10 last portfolio returns on the CEF-ARPO. The average APLs for the three scenarios are 3.324%, -0.109% and 3.274%, respectively (See also Table B.10, where the average APL, average computational times and the number of infeasible rates of return are summarized). Thus our algorithm delivers a reasonable-quality solution also in an environment that allows for inaccurate inputs. However, these results should be interpreted with caution. Importantly, it should be recognized that S1 – S3 amount to different optimization problems. A relative larger APL for S1 may imply that it is more costly – in terms of the portfolio variance – to attain a given portfolio return when individual asset returns are randomized than when they are not. The induced random variation in individual asset returns effectively boosts the corresponding elements of the variance and covariance matrix. This leads to a larger overall portfolio variance for a given portfolio return. As a result, portfolios on the higher end of the efficient frontier become more costly and even

TABLE B.7: Comparison of results for different scenarios based on Instance 1 (Hang Seng) varying the returns and the covariance matrix.

Curve Position	Required Return	ARPO – Variance S0	ARPO – Variance S1	APL S1 – S0	Time (s)	ARPO – Variance S2	APL S2 – S0	Time (s)	ARPO – Variance S3	APL S3 – S0	Time (s)
20	0.002861137	0.000642411	0.000642777	0.057%	0.848	0.000643991	0.246%	0.194	0.000644237	0.284%	1.991
40	0.002941981	0.000642907	0.000643518	0.095%	0.686	0.000644294	0.216%	0.133	0.000644828	0.299%	0.182
60	0.003022827	0.000643746	0.000644209	0.072%	0.227	0.000644951	0.187%	0.061	0.000645441	0.263%	0.105
80	0.003103671	0.000644392	0.000645228	0.130%	0.102	0.000645594	0.186%	0.324	0.000646286	0.294%	0.030
100	0.003184516	0.000645472	0.000646467	0.154%	0.189	0.00064646	0.153%	0.154	0.000647432	0.304%	0.072
120	0.003265361	0.000646778	0.000647925	0.177%	0.026	0.000647665	0.137%	1.397	0.000648805	0.313%	0.035
140	0.003346206	0.000648311	0.000649556	0.192%	0.111	0.000649104	0.122%	0.224	0.000650406	0.323%	0.140
160	0.003427051	0.000649973	0.000651174	0.185%	0.413	0.000650778	0.124%	0.025	0.000652234	0.348%	0.050
180	0.003507896	0.000651665	0.000653049	0.212%	0.071	0.000652687	0.157%	0.249	0.000654075	0.370%	0.143
200	0.00358874	0.000653615	0.000655182	0.240%	0.228	0.000654567	0.146%	0.043	0.000655994	0.364%	0.369
1820	0.010137479	0.003577353	0.003846626	7.527%	0.000	0.003553484	-0.667%	0.003	0.00382672	6.971%	0.006
1840	0.010218315	0.003690754	0.003973414	7.659%	0.003	0.003668539	-0.602%	0.003	0.003955413	7.171%	0.005
1860	0.010299151	0.003809087	0.00410529	7.776%	0.001	0.003788622	-0.537%	0.000	0.004089294	7.356%	0.000
1880	0.010379986	0.003932352	0.004242251	7.881%	0.000	0.003913731	-0.474%	0.002	0.004228359	7.527%	0.000
1900	0.010460822	0.004060548	0.004384302	7.973%	0.019	0.00404387	-0.411%	0.004	0.004372613	7.685%	0.003
1920	0.010541657	0.004193676	0.004531439	8.054%	0.008	0.004179034	-0.349%	0.001	0.004522052	7.830%	0.001
1940	0.010622493	0.004331735	0.004683665	8.124%	0.002	0.004319228	-0.289%	0.003	0.004676680	7.963%	0.000
1960	0.010703329	0.004474726	NF			0.00446445	-0.230%	0.000	NF		
1980	0.010784164	0.004622648	NF			0.004614697	-0.172%	0.004	NF		
2000	0.010865	0.004775501	NF			0.004769974	-0.116%	0.000	NF		

Notes: Table 7 summarizes the results of ARPO for different scenarios generated from Instance 1 (Hang Seng), in which the returns and the variance-covariance matrix are randomized. SX represents the scenario number X, where S0 is the benchmark scenario. NF indicates non-feasible portfolios.

infeasible (See also Table 10). In S2, where individual variances and covariances are randomized, the resulting CEF-ARPO remains very similar to the CEF-ARPO, as manifested by the average APL. Indeed, in contrast to S1, adding a normally distributed random disturbance term to variances and covariances between individual assets does not appear to alter the optimal solution. For portfolio on the lower end of the efficient frontier, the APL takes on low positive values; however, for more rewarding and riskier portfolios with feasible returns, the solution is better than for the CEF-ARPO, suggesting further diversification opportunities for investors in high-yield portfolios. Finally, S3 shows combined effects of S1 and S2. Next, we carry out a stability analysis of the ARPO algorithm to ratify our main findings. The stability analysis consists of six scenarios (numbered successively) as follows, which are evaluated against S0.

- Scenario 4 (S4): the minimum number of assets in the portfolio is increased from 1 to 2.
- Scenario 5 (S5): the maximum number of assets in the portfolio is decreased from 10 to 9.
- Scenario 6 (S6): Scenarios S4 and S5 are jointly considered.
- Scenario 7 (S7): the minimum quantity for all assets is increased from 0.01 to 0.015.
- Scenario 8 (S8): the maximum quantity for all assets is decreased from 1 to 0.995
- Scenario 9 (S9): Scenarios S7 and S8 are jointly considered.

Observe that in Scenarios S4 – S9 the vector of asset returns and the variance and covariance matrix are calculated from historical data and, in contrast to Scenarios S1 – S3, do not involve any random inputs. Table 9 presents the calculated CEF-ARPO variances and APLs for Scenarios S4 – S6. The calculations show that increasing the lower bound of the cardinality constraint does not have any effect on APLs or computational times. The APL is always 0 irrespectively of the CEF-ARPO position, including the higher end of the CEF. Similarly, decreasing the number of assets from 10 to 9 under S5 poses no significant challenge to portfolio diversification, since the resulting APLs – mainly at the lower end of the CEF – are very low. Scenario S6 that combines the two preceding scenarios shows some insignificant departures from S0, essentially due to the lower maximum number of assets that can be included in a portfolio.

Finally, Scenarios S7 – S9 show that small variations in the minimum and maximum quantities are followed by similarly unimportant changes in the CEF. Indeed, an increase in the minimum weight attributed to each asset leads to the average APL of 0.013% (S7 and S9) relative to the optimal solution under S0 (See also Table B.10). However, a decrease in the maximum weight does not appear to alter the optimal solution. All in all, additional computational experiments indicate that the ARPO algorithm shows a reasonable level of stability when dealing with constrained portfolio optimization problems.

7. Conclusions and future work

In this paper we propose the ARPO algorithm to solve the constrained mean-variance optimization problem. The specific constraints we use include the cardinality and quantity constraints. The ensuing complexity of the problem rules out closed-form solutions and conventional optimization methods. Specifically, the cardinality and quantity constraints render the problem computationally expensive to solve large-scale instances. Therefore, the use of metaheuristics –that handle such problems more flexibly and efficiently– is needed. The ARPO algorithm first generates an initial feasible solution. Then, this solution is iteratively improved using an iterated local search process which combines quadratic programming with a cache of previously computed solutions. The quality of the best-found solution is evaluated using the average percentage loss relative to the unconstrained efficient frontier.

According to the computational experiments, our solver outperforms –in terms of average percentage loss– most of the recent state-of-art approaches used in the literature, which tend to be more complex and difficult to implement in practical applications. In addition, the running time for our solver is notably shorter than the solvers used in other recent studies.

In future research, tighter cardinality and quantity constraints should be considered, as well as the pre-assignment constraint. Mean-variance portfolio optimization provides a natural laboratory for testing the performance of ARPO. While this algorithm is tested on the five benchmark

TABLE B.8: Comparison of results for different scenarios based on Instance 1 (Hang Seng) varying problem cardinality constraints.

Curve Position	Required Return	ARPO – Variance S0	ARPO – Variance S4	APL S4 – S0	Time (s)	ARPO – Variance S5	APL S5 – S0	Time (s)	ARPO – Variance S6	APL S6 – S0	Time (s)
20	0.002861137	0.000642411	0.000642411	0.000%	1.446	0.00064252	0.017%	0.557	0.00064252	0.017%	0.254
40	0.002941981	0.000642907	0.000642907	0.000%	0.118	0.000643026	0.019%	0.368	0.000643026	0.019%	0.391
60	0.003022827	0.000643746	0.000643746	0.000%	0.127	0.000643875	0.020%	0.527	0.000643875	0.020%	0.256
80	0.003103671	0.000644392	0.000644392	0.000%	0.365	0.000645066	0.105%	0.185	0.000645066	0.105%	0.090
100	0.003184516	0.000645472	0.000645472	0.000%	0.338	0.0006466	0.175%	0.191	0.0006466	0.175%	0.285
120	0.003265361	0.000646778	0.000646778	0.000%	0.012	0.000647619	0.130%	0.803	0.000647619	0.130%	0.058
140	0.003346206	0.000648311	0.000648311	0.000%	0.101	0.000648891	0.089%	0.171	0.000648891	0.089%	0.138
160	0.003427051	0.000649973	0.000649973	0.000%	0.168	0.000650437	0.071%	0.123	0.000650437	0.071%	0.020
180	0.003507896	0.000651665	0.000651665	0.000%	0.589	0.000652259	0.091%	0.408	0.000652259	0.091%	0.131
200	0.00358874	0.000653615	0.000653615	0.000%	0.089	0.000654354	0.113%	0.141	0.000654354	0.113%	0.093
1820	0.010137479	0.003577353	0.003577352	0.000%	0.002	0.003577352	0.000%	0.000	0.003577352	0.000%	1.847
1840	0.010218315	0.003690754	0.003690754	0.000%	4.421	0.003690754	0.000%	0.002	0.003690754	0.000%	3.000
1860	0.010299151	0.003809087	0.003809088	0.000%	1.707	0.003809088	0.000%	0.002	0.003809088	0.000%	2.188
1880	0.010379986	0.003932352	0.003932352	0.000%	1.766	0.003932352	0.000%	0.001	0.003932352	0.000%	0.743
1900	0.010460822	0.004060548	0.004060549	0.000%	2.067	0.004060549	0.000%	0.002	0.004060549	0.000%	0.118
1920	0.010541657	0.004193676	0.004193675	0.000%	0.987	0.004193675	0.000%	0.048	0.004193675	0.000%	0.507
1940	0.010622493	0.004331735	0.004331735	0.000%	14.906	0.004331735	0.000%	0.001	0.004331735	0.000%	0.685
1960	0.010703329	0.004474726	0.004474726	0.000%	0.000	0.004474726	0.000%	0.000	0.004474726	0.000%	0.000
1980	0.010784164	0.004622648	0.004622647	0.000%	0.000	0.004622647	0.000%	0.001	0.004622647	0.000%	0.000
2000	0.010865	0.004775501	0.004775501	0.000%	0.000	0.004775501	0.000%	0.000	0.004775501	0.000%	0.000

Notes: Table 8 summarizes the results of ARPO for different scenarios generated from Instance 1 (Hang Seng), in which problem cardinality constrains are modified. SX represents the scenario number X, where S0 is the benchmark scenario.

TABLE B.9: Comparison of results for different scenarios based on Instance 1 (Hang Seng) varying problem quantity constraints.

Curve Position	Required Return	ARPO – Variance S0	ARPO – Variance S7	APL S7 – S0	Time (s)	ARPO – Variance S8	APL S8 – S0	Time (s)	ARPO – Variance S9	APL S9 – S0	Time (s)
20	0.002861137	0.000642411	0.00064252	0.017%	0.953	0.000642411	0.000%	0.701	0.00064252	0.017%	0.174
40	0.002941981	0.000642907	0.000643026	0.019%	0.962	0.000642907	0.000%	0.387	0.000643026	0.019%	0.094
60	0.003022827	0.000643746	0.000643875	0.020%	0.087	0.000643746	0.000%	0.789	0.000643875	0.020%	0.229
80	0.003103671	0.000644392	0.000645066	0.105%	0.073	0.000644392	0.000%	0.144	0.000645066	0.105%	0.424
100	0.003184516	0.000645472	0.000646032	0.087%	1.309	0.000645472	0.000%	0.246	0.000646032	0.087%	0.044
120	0.003265361	0.000646778	0.000646778	0.000%	0.154	0.000646778	0.000%	0.236	0.000646778	0.000%	0.031
140	0.003346206	0.000648311	0.000648311	0.000%	0.083	0.000648311	0.000%	0.005	0.000648311	0.000%	0.104
160	0.003427051	0.000649973	0.00065007	0.015%	0.420	0.000649973	0.000%	0.642	0.00065007	0.015%	0.032
180	0.003507896	0.000651665	0.000651665	0.000%	0.681	0.000651665	0.000%	0.388	0.000651665	0.000%	0.148
200	0.00358874	0.000653615	0.000653615	0.000%	0.372	0.000653615	0.000%	0.282	0.000653615	0.000%	0.109
1820	0.010137479	0.003577353	0.003577352	0.000%	0.006	0.003577352	0.000%	0.000	0.003577352	0.000%	0.002
1840	0.010218315	0.003690754	0.003690754	0.000%	0.006	0.003690754	0.000%	0.002	0.003690754	0.000%	0.001
1860	0.010299151	0.003809087	0.003809088	0.000%	0.000	0.003809088	0.000%	0.001	0.003809088	0.000%	0.000
1880	0.010379986	0.003932352	0.003932352	0.000%	0.001	0.003932352	0.000%	0.000	0.003932352	0.000%	0.002
1900	0.010460822	0.004060548	0.004060549	0.000%	0.001	0.004060549	0.000%	0.011	0.004060549	0.000%	0.000
1920	0.010541657	0.004193676	0.004193675	0.000%	0.010	0.004193675	0.000%	0.002	0.004193675	0.000%	0.000
1940	0.010622493	0.004331735	0.004331735	0.000%	0.001	0.004331735	0.000%	0.000	0.004331735	0.000%	0.003
1960	0.010703329	0.004474726	0.004474726	0.000%	0.000	0.004474726	0.000%	0.000	0.004474726	0.000%	0.003
1980	0.010784164	0.004622648	0.004622647	0.000%	0.000	0.004622647	0.000%	0.001	0.004622647	0.000%	0.004
2000	0.010865	0.004775501	0.004775501	0.000%	0.000	0.004775501	0.000%	0.000	0.004775501	0.000%	0.000

Notes: Table 9 summarizes the results of ARPO for different scenarios generated from Instance 1 (Hang Seng), in which problem quantity constrains are modified. SX represents the scenario number X, where S0 is the benchmark scenario.

TABLE B.10: Different scenarios based on Instance 1 (Hang Seng).

Scenario	Average APL	Non feasible portfolios	Time (s)
Scenario 1	3.324%	3	0.173
Scenario 2	-0.109%	0	0.141
Scenario 3	3.274%	3	0.196
Scenario 4	0.000%	0	1.460
Scenario 5	0.042%	0	0.177
Scenario 6	0.042%	0	0.540
Scenario 7	0.013%	0	0.256
Scenario 8	0.000%	0	0.192
Scenario 9	0.013%	0	0.070

Notes: Table 10 shows the average APL (column 2), the number of non feasible portfolios (column 3), and the average computational times (column 4) obtained for 9 different scenarios on Instance 1 (Hang Seng).

indices, an unexhausted list application could potentially include other periods, different countries, regions, sectors and asset classes. Moreover, the practical appeal of ARPO is not limited to portfolio optimization, but rather could be extended to other optimization problem comprising, for instance, asset and liability management of companies. Likewise, it can be conveniently applied to construct a portfolio of international investments and to study whether ARPO can alleviate the widely documented “home bias" phenomenon.

Investment portfolio formulation with variance as a risk measure is just one option. It is possible to extend presented research using multi-criteria investing portfolio models (Sawik, 2012b; Sawik, 2012a) with the use of value-at-risk (VaR) and conditional value-at-risk (CVaR) or multi-objective supply portfolio models (Sawik, 2011; Sawik, 2013b) with VaR and CVaR or with differently formulated risk measures (Heckmann et al., 2015). In all these cases a novel contribution to this issue would consist in a comparison between exact methods and ARPO-like metaheuristics.

One important shortcoming in the literature of mean-variance portfolio optimization is that the models typically rely upon expected returns and variance data, instead of considering the return as a real random variable. A novel contribution to this issue would consist of adding stochastic elements to the problem formulation, so that it includes random effects.

Acknowledgements

This work has been partially supported by the Spanish Ministry of Economy and Competitiveness (TRA2013-48180-C3-3-P, TRA2015-71883-REDT) and FEDER. Likewise we want to acknowledge the support received by the Department of Universities, Research & Information Society of the Catalan Government (Grant 2014-CTP-00001) and by NCN grant (DEC-2013/11/B/ST8/04458).

References

Angelelli, Enrico, Renata Mansini, and M Grazia Speranza (2007). “A comparison of MAD and CVaR models with real features”. In: *Journal of Banking Finance* 32.7, pp. 1188–1197.

Armañanzas, R. and J. A. Lozano (2005). “A multiobjective approach to the portfolio optimization problem”. In: *IEEE Congress on Evolutionary Computation*. Vol. 2. IEEE Press, pp. 1388–1395.

Bertsimas, Dimitris and Dessislava Pachamanova (2008). “Robust multiperiod portfolio management in the presence of transaction costs”. In: *Computers and Operations Research* 35.1, pp. 3–17.

Bienstock, D. (1996). “Computational study of a family of mixed-integer quadratic programming problems”. In: *Mathematical programming* 74.2, pp. 121–140.

Blum, Christian, Jakob Puchinger, Günther R Raidl, and Andrea Roli (2011). “Hybrid metaheuristics in combinatorial optimization: A survey”. In: *Applied Soft Computing* 11.6, pp. 4135–4151.

Blume, M. E. and I. Friend (1975). "The asset structure of individual portfolios and some implications for utility functions". In: *Journal of Finance* 30.2, pp. 585–603.

Boussaïd, I., J. Lepagnot, and P. Siarry (2013). "A survey on optimization metaheuristics". In: *Information Sciences* 237, pp. 82–117.

Bruni, R., F. Cesarone, A. Scozzari, and F. Tardella (2015). "A linear risk-return model for enhanced indexation in portfolio optimization". In: *OR Spectrum* 37.3, pp. 735–759.

Burke, E. K., M. Hyde, G. Kendall, G. Ochoa, E. Özcan, and J. R. Woodward (2010). "A classification of hyper-heuristic approaches". In: *Handbook of metaheuristics*, pp. 449–468.

Chang, T.-J., N. Meade, J. E. Beasley, and Y. M. Sharaiha (2000). "Heuristics for cardinality constrained portfolio optimisation". In: *Computers & Operations Research* 27.13, pp. 1271–1302.

Crama, Yves and Michaël Schyns (2003). "Simulated annealing for complex portfolio selection problems". In: *European Journal of Operational Research* 150.3, pp. 546–571.

DeMiguel, V., L. Garlappi, and R. Uppal (2009). "Optimal versus naïve diversification: How inefficient is the 1/N portfolio strategy?" In: *Review of Financial Studies* 22.5, pp. 1915–1953.

Di Gaspero, L., G. Di Tollo, A. Roli, and A. Schaerf (2011). "Hybrid metaheuristics for constrained portfolio selection problems". In: *Quantitative Finance* 11.10, pp. 1473–1487.

Di Tollo, G. and A. Roli (2008). "Metaheuristics for the portfolio selection problem". In: *International Journal of Operations Research* 5.1, pp. 13–35.

Elton, E. J., M. J. Gruber, and M. W. Padberg (1976). "Simple criteria for optimal portfolio selection". In: *Journal of Finance* 31.5, pp. 1341–1357.

Evans, J. L. and S. H. Archer (1968). "Diversification and the reduction of dispersion: an empirical analysis". In: *Journal of Finance* 23.5, pp. 761–767.

Fernández, A. and S. Gómez (2007). "Portfolio selection using neural networks". In: *Computers and Operations Research* 34.4, pp. 1177–1191.

Fujii, M. and A. Takahashi (2014). "Making mean-variance hedging implementable in a partially observable market". In: *Quantitative Finance* 14.10, pp. 1709–1724.

Guiso, Luigi, Tullio Jappelli, and Daniele Terlizzese (1996). "Income risk, borrowing constraints, and portfolio choice". In: *The American Economic Review* 86.1, pp. 158–172.

Heckmann, Iris, Tina Comes, and Stefan Nickel (2015). "A critical review on supply chain risk – Definition, measure and modeling". In: *Omega. The International Journal of Management Science* 52, pp. 119–132.

Jain, A., P. K. Jain, T. H. McInish, and M. McKenzie (2013). "Worldwide reach of short selling regulations". In: *Journal of Financial Economics* 109.1, pp. 177–197.

Jansen, R. and R. van Dijk (2002). "Optimal benchmark tracking with small portfolios". In: *Journal of Portfolio Management* 28.2, pp. 9–22.

Jobst, Norbert J, Michael D Horniman, Cormac A Lucas, and Gautam Mitra (2001). "Computational aspects of alternative portfolio selection models in the presence of discrete asset choice constraints". In: *Quantitative finance* 1.5, pp. 489–501.

Jorion, Philippe (1985). "International portfolio diversification with estimation risk". In: *Journal of Business* 58.3, pp. 259–278.

Jourdan, L., M. Basseur, and E.-G. Talbi (2009). "Hybridizing exact methods and metaheuristics: A taxonomy". In: *European Journal of Operational Research* 199.3, pp. 620–629.

Juan, A. A., J. Faulin, R. Ruiz, B. Barrios, and S. Caballe (2010). "The SR-GCWS hybrid algorithm for solving the capacitated vehicle routing problem". In: *Applied Soft Computing* 10.1, pp. 215–224.

Juan, A. A., J. Faulin, J. Jorba, D. Riera, D. Masip, and B. Barrios (2011a). "On the use of Monte Carlo simulation, cache and splitting techniques to improve the Clarke and Wright savings heuristics". In: *Journal of the Operational Research Society* 62, pp. 1085–1097.

Juan, A. A., J. Faulin, A. Ferrer, H. Lourenço, and B. Barrios (2013a). "MIRHA: multi-start biased randomization of heuristics with adaptive local search for solving non-smooth routing problems". In: *TOP* 21, pp. 109–132.

Kallberg, J. G. and W. T. Ziemba (1984). "Mis-specifications in portfolio selection problems". In: *Risk and Capital*. Springer-Verlag, New York, pp. 74–87.

Kolm, P. N., R. Tütüncü, and F. J. Fabozzi (2014). "60 years of portfolio optimization: Practical challenges and current trends". In: *European Journal of Operational Research* 234.2, pp. 356–371.

Leippold, M., F. Trojani, and P. Vanini (2011). "Multiperiod mean-variance efficient portfolios with endogenous liabilities". In: *Quantitative Finance* 11.10, pp. 1534–1546.

Levy, M. and Y. Ritov (2011). "Mean–variance efficient portfolios with many assets". In: *Quantitative Finance* 11.10, pp. 1461–1471.

Lloyd, W. P., J. H. Hand, and N. K. Modani (1981). "The effect of portfolio construction rules on the relationship between portfolio size and effective diversification". In: *Journal of Financial Research* 4.3, pp. 183–193.

Lourenço, H.R., O.C. Martin, and T. Stützle (2010). "Iterated local search: framework and applications". In: *Handbook of Metaheuristics*. Ed. by M. Gendreau and J.-Y. Potvin. Vol. 146. International Series in Operations Research & Management Science. Springer US, pp. 363–397.

Mansini, R., W. Ogryczak, and M. G. Speranza (2014). "Twenty years of linear programming based portfolio optimization". In: *European Journal of Operational Research* 234.2, pp. 518–535.

Maringer, D. and H. Kellerer (2003). "Optimization of cardinality constrained portfolios with a hybrid local search algorithm". In: *OR Spectrum* 25.4, pp. 481–495.

Markowitz, H. (1952). "Portfolio selection". In: *The journal of finance* 7.1, pp. 77–91.

Michaud, R. O. (1989). "The Markowitz optimization enigma: is 'optimized' optimal?" In: *Financial Analysts Journal* 45.1, pp. 31–42.

Moral-Escudero, R., R. Ruiz-Torrubiano, and A. Suárez (2006). "Selection of optimal investment portfolios with cardinality constraints". In: *2006 IEEE International Conference on Evolutionary Computation*. IEEE, pp. 2382–2388.

Nawrocki, David N (1983). "A comparison of risk measures when used in a simple portfolio selection heuristic." In: *Journal of Business Finance Accounting* 10.2, pp. 183–194.

— (2000). "Portfolio Optimization, Heuristics and the 'Butterfly Effect'". In: *Journal of Financial Planning-Denver* 13.2, pp. 68–75.

Ólafsson, S. (2006). "Chapter 21 Metaheuristics". In: *Handbooks in Operations Research and Management Science 13*. Elsevier, pp. 633–654.

Pachamanova, D. A. and F. J. Fabozzi (2010). *Simulation and Optimization in Finance*. John Wiley and Sons.

Pae, Y. and N. Sabbaghi (2014). "Log-robust portfolio management after transaction costs". In: *OR Spectrum* 36.1, pp. 95–112.

Perez, G. B., D. F. Jones, M. Tamiz, and T. A. Bilbao (2007). "An interactive three-stage model for mutual funds portfolio selection". In: *Omega. The International Journal of Management Science* 35.1, pp. 75–88.

Pınar, M. (2007). "Robust scenario optimization based on downside-risk measure for multi-period portfolio selection". In: *OR Spectrum* 29.2, pp. 295–309.

Sawik, B. (2012a). "Bi-criteria portfolio optimization models with percentile and symmetric risk measures by mathematical programming". In: *Przeglad Elektrotechniczny* 88.10B, pp. 176–180.

— (2012b). "Downside risk approach for multi-objective portfolio optimization". In: *Operations Research Proceedings*. Ed. by D. Klatte, H.J. Luthi, and K. Schmedders. Springer, pp. 191–196.

— (2013a). "Survey of multi-objective portfolio optimization by linear and mixed integer programming". In: *Applications of Management Science*. Ed. by K. D. Lawrence and G. Kleinman. Vol. 16. Emerald Group Publishing Limited, pp. 55–79.

Sawik, T. (2011). "Selection of supply portfolio under disruption risks". In: *Omega. The International Journal of Management Science* 39.2, pp. 194–208.

— (2013b). "Selection of resilient supply portfolio under disruption risks". In: *Omega. The International Journal of Management Science* 41.2, pp. 259–269.

Schaerf, A. (2002). "Local search techniques for constrained portfolio selection problems". In: *Computational Economics* 20.3, pp. 177–190.

Sharpe, W. F. (1967). "A linear programming algorithm for mutual fund portfolio selection". In: *Management Science* 13.7, pp. 499–510.

— (1971). "A linear programming approximation for the general portfolio analysis problem". In: *Journal of Financial and Quantitative Analysis* 6.05, pp. 1263–1275.

Speranza, M Grazia (1993). "Linear programming models for portfolio optimization". In: 14, pp. 107–123.

Tobin, J. (1958). "Liquidity preference as behavior towards risk". In: *Review of Economic Studies* 25.2, pp. 65–86.

Tobin, James (1965). "The theory of portfolio selection". In: *The theory of interest rates*. Ed. by F. Hahn and F. Brechling. Macmillan & Co. Ltd., London, pp. 3–51.

Vigna, E. (2014). "On efficiency of mean–variance based portfolio selection in defined contribution pension schemes". In: *Quantitative Finance* 14.2, pp. 237–258.

Wolfe, P. (1959). "The simplex method for quadratic programming". In: *Econometrica* 27, pp. 382–398.

Zumbach, G. (2013). "A mean/variance approach to long-term fixed-income portfolio allocation". In: *Quantitative Finance* 13.9, pp. 1459–1471.

B.8 Sustainable Urban Freight Transport: a multi-depot vehicle routing problem considering different cost dimensions

Lorena Reyes[1], Laura Calvet[2], Carla Talens[2], Angel A. Juan[2], Javier Faulin[1]

1. Statistics and Operations Research Department, Public University of Navarra, Pamplona, Spain
e-mail: {lorena.reyes,javier.faulin}@unavarra.es
2. Computer Science Department, Open University of Catalonia – IN3, Castelldefels, Spain
e-mail: {lcalvetl,ctalens,ajuanp}@uoc.edu

Abstract

Urban freight transport is becoming increasingly complex due to an increase in the number of journeys, and the associated volume and frequency. In addition, stakeholders' preferences and city logistics dynamics affect the freight flow and the distribution process efficiency in downtown. In general, transport activities have a significant negative impact on the environment and population welfare, which motivates decision-makers to study the transport efficiency from a sustainability perspective. This work proposes a metaheuristic-based approach for tackling the multi-depot vehicle routing problem with the goal of minimizing the distribution costs considering sustainability, i.e., economic, environmental and social impacts. A set of computational experiments are carried out to illustrate the problem, prove the need of an approximate approach, test it, show suitable visualization techniques and analyze how the objective values change under different scenarios.
Keywords: sustainability, transport, routing, multi-depot vehicle routing problem, metaheuristics.

1. Introduction

According to Eurostat (2015), emissions of greenhouses gases, air pollutants and noise from transport affect the climate, environment and human health. Given the importance of these facts, the EU sustainable development strategy defines sustainable transport as one of its seven key challenges. In this context, the increasing social concern is compelling companies to change purely commercial objectives in order to consider sustainability. This new vision seeks to compensate the negative impacts of transport activities without neglecting economic profits. Despite the fact that the literature on transport is extensive, there is a lack of works on urban transport taking into account social and environmental issues.

A relevant characteristic in urban freight transport is the existence of patterns recognized as off-peak hours and peak hours. The latter cover intervals of time in which a large number of commercial activities are carried out, including starting and ending work shifts. In fact, many companies and shopkeepers establish similar time windows, which results in short operation times. As a consequence, goods and individual transport is concentrated in specific peak hours, generating traffic congestion (Muñuzuri et al., 2010). Numerous indicators to characterize urban transport have been proposed during the recent years. For example, the access probability for a specific area is used for route scheduling: if it is low, companies are forced to increase the number of vehicles on roads to complete planned deliveries (Hunt and Stefan, 2007). Another indicator is the mean speed, which reflects the traffic status. It is also important to consider information regarding speed variations, which generate economic and environmental impacts due to high fuel consumption, long operation times and pollution. Moreover, high variations are associated to a high accidentally risk for both pedestrian and vehicles (Wang et al., 2016). In the same line, Xie et al. (2013) affirm that the probability of traffic accidents is influenced by the number and the frequency of traffic signs, which affect speed variations and the mean speed.

Shippers need to design distribution routes to pick-up and/or deliver goods to customers. Frequently, urban zones are crowded and congested following the aforementioned patterns (peak-hours and off-peak hours), which severely affects distribution plans. These characteristics hinder constant and smooth freight flows due to the speed variations and idle times and, in turn, by the reduced accessibility to specific zones (Chen et al., 2013). An urban distribution network integrates production plants usually located in city outskirt, distribution points that geographically are intermediate points, and shopkeepers' facilities located at downtown or urban zones. Consequently,

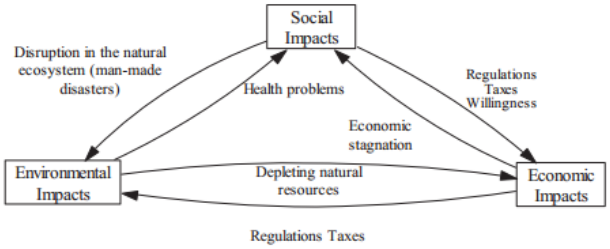


FIGURE B.1: Negative impacts' influence (based on McKinnon et al., 2015).

distribution processes imply crossing a city from production plants to customer facilities. Besides, current market trends such as e-commerce have made that freight distribution processes involve also residential zones, thus changing logistics networks. The expansion of e-commerce obliges producers to consider not only the transport of high amounts to commercial establishments, but also the pick-up and/or delivery of small packages to end customers (Ruan et al., 2012). On the one side, this type of trade reduces the traffic congestion because some customers may replace shopping centers by internet shopping. On the other side, e-commerce is a market with a high uncertainty, generating a disaggregated and scattered demand which may increase the negative impacts of urban transport (Teo et al., 2012). Thus, advanced optimization techniques are required to design routes in these increasingly challenging scenarios.

Boosted by social pressures, sustainability pillars are starting to be considered decision criteria in distribution processes. While economic impacts can be measured through increases in operational costs, social and environmental assessments tend to be subjective. Usually, these assessments have an economic perspective in order to provide a dimension of the impacts, and to take them into account in the decision-making (Ranaiefar and Amelia, 2011). Figure B.1 shows how economic, environmental and social impacts are interrelated. Thus, it is important to consider all dimensions. Prevention and mitigation costs generated by negative impacts should appear in financial reports. Prevention costs are due to the economic regulations associated to natural resources consumption or pollutant emissions, and are imposed by governments to avoid or minimize social and environmental consequences. Regarding mitigation costs, they are related to penalties for generating more emissions than the allowable (Santos et al., 2010). In addition, companies incur in prevention or mitigation costs by implementing sustainable strategies such as developing alternative materials and more sustainable production processes (Scheuer, 2005).

This work focuses on the distribution process in urban zones considering multiple capacitated depots. We propose an approach to solve the multi-depot vehicle routing problem (MDVRP) minimizing the distribution costs, which depend on multiple sustainability indicators/criteria. These indicators are: traveling time and distance, carbon emissions, and risk of accidents (i.e., social impact). Note that these indicators may conflict; for instance, minimizing traveling time and distance may not lead to the same solution in a scenario with congestion. Figure B.2 illustrates the complexity of achieving a balance between the economic, social and environmental impacts. Nodes' size reveals the demand (i.e., the bigger the circle, the higher the demand). While minimizing carbon emissions and risk of accidents depends on the traveling distance, they are also affected by the load of the vehicles, which is totally ignored when optimizing traveling distance. Being an extension of the classical vehicle routing problem (VRP), the MDVRP is also $\mathcal{NP}$-hard. As a consequence, an algorithm based on heuristics or metaheuristics (Talbi, 2009) is required to obtain high-quality solutions in a reasonable computing time. The solving approach we propose relies on the variable neighborhood search (VNS) metaheuristic (Gendreau and Potvin, 2010), and the construction of solutions is based on the biased randomization version (Juan et al., 2011a) of the classical Clarke and Wright's savings (CWS) heuristic (Clarke and Wright, 1964). A mixed-integer mathematical formulation is presented to define the problem and get optimal solutions for some instances. This experiment evidences the need of a metaheuristic-based approach to solve instances with a realistic size. More computational experiments are performed adapting benchmark MDVRP instances to gain insights into the problem and the relationship between the sustainable indicators. In order to efficiently represent the solutions found using multiple criteria and provide decision-makers with a tool to compare them, we propose the design of radar plots differentiating relevant regions. An example with different solutions is shown in Figure B.3. Each axis represents an indicator and ranges from the minimum value found to the maximum one. The small pentagon linking all the

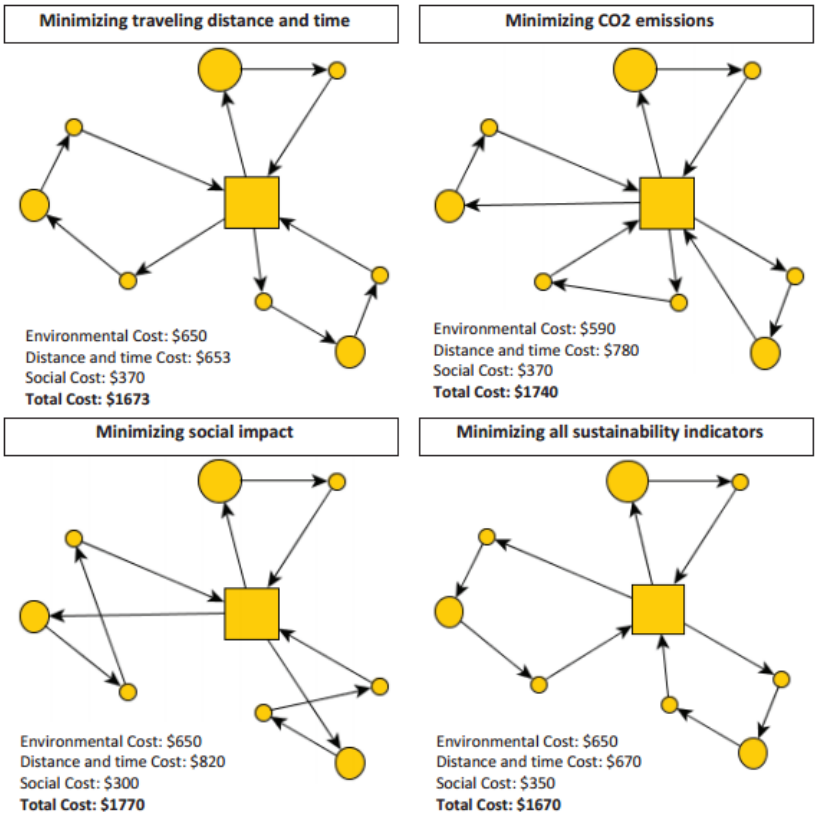


FIGURE B.2: Representation of MDVRP (with a single depot) solutions considering different criteria.

minimum values describes the behavior of the "ideal", and probably inexistent, solution, which achieves the minimum value for each indicator. Assuming that the solutions found are optimal, the area inside the pentagon represents a solution subspace that cannot be reached. The desirable region is defined as the union of the subspaces associated to the solutions. Solutions falling outside the figure may be discarded since they are dominated by at least one solution (i.e., there is at least one solution with the same or better values for all indicators). Similarly, the intersections of at least two solution subspaces form the sustainability region, which contains the best solutions, i.e., solutions that achieve a suitable balance considering a number of indicators. To the best of our knowledge, this is the first work addressing a rich VRP (RVRP) with sustainability indicators.

The rest of the paper is structured as follows: next section provides a literature review. Section 3 offers a detailed description of the problem analyzed, including a mathematical formulation. Section 4 proposes a solving approach based on a metaheuristic. The computational experiments are explained in Section 5, while Section 6 discusses the results. Finally, Section 7 gathers the conclusions and identifies lines for future research.

2. Literature review

The increasing social concern for the environment and a sustainable growth in general requires the transformation of cities. In this context, the classical VRP may be enriched to include characteristics that allow the reduction of environmental and social impacts in urban zones concerning transport activities. During the last decade, this problem has been complimented by a large number of variants including: the green VRP (GVRP) and the pollution VRP (PVRP). While the former is focused on the environmental impact caused by the fuel or energy consumption of transport, the latter takes into account the pollution and different emissions generated. Thus, both problems analyze the emissions and fuel/energy consumption levels, which depend on traffic congestion, speed, acceleration, type of road, type of vehicle, and load, among other internal and external factors of the operation (Bektaş and Laporte, 2011; Koç et al., 2014). A large number of models are considered RVRPs, which encompass special characteristics from city logistics and smart cities,

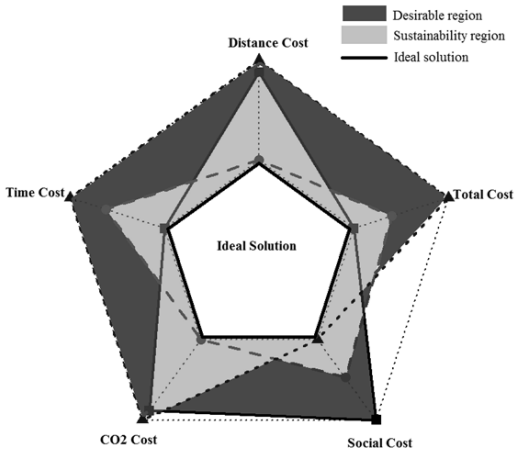


FIGURE B.3: Representation of different solutions and regions.

for instance, the integration of information tools and communication (ITC) in transport operations. The reader interested in a comprehensive review on realistic and rich variants of the VRP is referred to Caceres et al. (2014).

Regarding environmental impacts, the distance and vehicle weight play a crucial role in the fuel/energy consumption and carbon emissions, thereby Ubeda et al. (2011) aimed at reducing transport costs and emissions, considering the distance and some variations in the vehicle maximum capacity. It is concluded that enhancing load factors (which may be achieved by using heterogeneous fleets) is an efficient way to get significant savings and environmental benefits. The authors also discuss negative externalities of transport such as noise, air pollution, congestion, accident rate, energy consumption and land use, among others. There are studies tackling the negative impacts from three different perspectives: negative externalities, emissions released and fuel consumption. Faulin et al. (2011), Liu et al. (2014), and Zhang et al. (2015) considered environmental indicators for the capacitated VRP; they affirm that the load variation defines fuel consumption and emissions caused by transport. Besides, the load variation influences the distribution processes profitability. In this line, Kuo (2010), Demir et al. (2014), and Xiao and Konak (2015) developed methodologies for the green heterogeneous VRP (green HVRP), considering traffic congestion, road gradient, speed variations and distance traveled as variables that influence fuel consumption and as elements that characterize the urban transport dynamics (Jabbarpour et al., 2015). More recently, Niknamfar and Niaki (2016) study the MDVRP with time windows to optimize the customers-depots allocation and the vehicles selection aiming to minimize the environmental impacts. They demonstrated that an optimal allocation and coordination between stakeholders not only reduce the negative impacts but also enhance the total profit. Juan et al. (2014e) considered a supply chain with multiple suppliers for minimizing the empty trips and the travel distance in each route. They concluded that it is possible to reduce the CO2 emission to 23% when the distribution process is carried out in collaboration with multiple suppliers. Wang et al. (2014b) demonstrated that considering environmental criteria allows a saving up to 10% of the operation costs. The authors developed an algorithm to integrate the economic and environmental goals based on the MDVRP with backhauls. Demir et al. (2014) considered the MDVRP with freight pick-up and delivery to ensure that any customer demand can be met from any depot and thus reducing the operation cost.

Some studies have focused on the analysis of environmental impacts caused by transport activities in urban zones, however there is no characterization for getting a rough estimation of the real impact of these activities. For instance, about 60% of transport activities take place in urban regions at where around 80% population is concentrated, making people the main harmed (European Commission, 2015). Social impact refers to health problems and other factors such as quietness, air quality, urban esthetic, accessibility and urban safety. Penalties, taxes or willingness to pay as a means to reduce the social impacts constitute the costs associated. It is estimated that about 0.4%, 0.2%, 1.5% and 2% of the gross domestic product is related to air pollution problems, noise, accidents and traffic congestion, respectively (Caceres et al., 2014). Therefore, the sustainability concept has started to take part in the decision-making process but there is a lack of structured

FIGURE B.4: Example of the potential lack of strong correlation between distance and traveling time.

tools that allow the integration of the three dimensions and support decision-makers (Chen et al., 2013).

There are only a few works on sustainability criteria. Chibeles-Martins et al. (2016) pose ecological criteria to determine an optimal structure of distribution networks. They solved a bi-objective problem focused on determining the suitable locations, capacities and attributes in factories, warehouses and a distribution center. The solution method is based on the simulating annealing metaheuristic and Pareto optimality is considered to get a balancing between economic and ecological concerns. In the same sense, Zhang et al. (2016) implement evolutionary algorithms to determine the optimal design of supply chains considering two possible scenarios: first, the transport is outsourced and second the transport is leased. It is a multi-objective problem aimed at minimizing CO2 emissions, fine dust and costs. The authors implement the non-dominated sorting genetic algorithm-II (NSGA-II) and the strength Pareto evolutionary algorithm2 (SEAP2) to compare their performance, both methods take into account Pareto optimality through a scalarization method computed by a weighted sum. Later, Kadziński et al. (2017) define a sustainable objective to design an optimal distribution structure considering a supply chain with multi-distribution channels. Objectives are maximizing customer coverage, and minimizing cost and environmental impacts. Notice that social objectives do not respond to problems highlighted by the society, besides these approaches belong to strategic levels without considering the synergy among tactical levels, operative levels and stakeholders' particular objectives.

3. Description of the problem

This paper studies a supply chain with multiple suppliers and customers, formulated as a MDVRP, in which the distribution process is carried out by a homogeneous fleet of capacitated vehicles. The problem consists on defining distribution routes considering the sustainability concept as optimality criteria. The three-axis of sustainability (measured as economic, environmental and social impacts) are represented by traveling distances and times, carbon emissions and risk of accidents. Several studies have addressed the economic impacts as a variable mainly influenced by traveling distances; therefore most existing models seek to minimize them. However, doing this does not guarantee the minimum impact because many elements such as congestion, speed limits, traffic signs and vehicles crashes make longer the time of the distribution routes (Wang et al., 2016). In fact, the shortest paths in urban zones tend to have more traffic signs since these are the most frequented and, as a result, main streets may be the slowest paths (see an illustration in Figure B.4). Accordingly, we also consider traveling times to represent these urban attributes.

Formally, the MDVRP can be defined as a graph $G=(N,A)$, where $N = \{N_d, N_c\}$ is a set of nodes, N_d and N_c representing the subsets of depots and customers respectively, and $A = \{(i, j) : i, j \in N, i \neq j\}$ is the set of arcs connecting all nodes in N. Each depot i ($\forall\ i \in N_d$) has a capacity s_i, and each customer j ($\forall\ j \in N_c$) has a demand $r_j (r_j \geq 0)$. The vehicle fleet K is composed of k identical vehicles ($K = \{1, ..., k\}$), and Q and D denote the capacity and the maximum distance

associated to each vehicle. Each arc $(i, j) \in A$ has associated a traveling distance (d_{ij}), traveling time (t_{ij}), fuel consumption (f_{ij}), and carbon emissions.

The binary variable x_{ijk} is employed to represent the routes: $x_{ijk} = 1$ if the arc (i, j) is traversed by vehicle k and $x_{ijk} = 0$ otherwise. The auxiliary variable U_{jk} is used for sub-tour elimination in route k. The binary variable z_{ij} indicates whether customer j is allocated to depot i ($z_{ij} = 1$) or not ($z_{ij} = 0$). In addition, flow variables y_{ijk} represent the load in the vehicle associated to the route k servicing customer j after visiting customer i.

Extension to consider sustainability indicators

The problem tackled considers the three-axis of sustainability.

- *Economic dimension:* It is composed by the classical measures total traveling times and distances, which are monetized based on the driver wage (DW), vehicle fixed cost (FC) and oil price (C_f). The cost of the total traveling times and distances per route are computed as follows:

$$\sum_{(i,j)\in N} (DW + FC) \cdot t_{ij} \cdot x_{ijk} \tag{B.1}$$

$$\sum_{(i,j)\in N} C_f \cdot f_{ij} \cdot x_{ijk} \tag{B.2}$$

- *Environmental dimension:* CO2 emissions estimates assume that the internal combustion process of vehicles burns the carbon of the fuel and it is released as carbon dioxide. Thus, emissions are assumed to depend on fuel consumption. Equation B.3 computes the cost of environmental impacts, considering a factor for carbon emissions (C_e).

$$\sum_{(i,j)\in N} C_e \cdot f_{ij} \cdot x_{ijk} \tag{B.3}$$

- *Social dimension:* Accidents are an externality caused by speed variations on roads, among other factors. These variations represent the state and stability of the roads, and are associated to an accident risk for pedestrian and vehicles (Wang et al., 2016). Equation B.4 represents the social cost, and depends on a given coefficient (a_{ij}), vehicle loading and traveling distance:

$$\sum_{(i,j)\in N} a_{ij} \cdot d_{ij} \cdot y_{ijk} \cdot x_{ijk} \tag{B.4}$$

The fuel consumption is estimated as suggested in Kuo (2010) and Zhang et al. (2015) (Equation B.5). lph_{ij} represents the fuel consumption per unit of time and p is a factor that penalize for each additional load (M). This value is determined by the average miles per fuel liter (kpl_{ij}) and velocity (v_{ij}) (Equation B.6). Without loss of generality, we will assume that p is equal to 0. Thus, f_{ijk} can be represented by f_{ij}.

$$f_{ijk} = lph_{ij} \cdot \frac{d_{ij}}{v_{ij}} \cdot \left(1 + p \cdot \frac{y_{ijk}}{M}\right) \quad \forall\, (i, j) \in N, k \in K \tag{B.5}$$

$$lph_{ij} = \frac{v_{ij}}{kpl_{ij}} \quad \forall\, (i, j) \in N \tag{B.6}$$

All in all, the objective is defined as a multi-criteria function considering the total traveling time, total traveling distance, environmental cost, and social cost.

$$\sum_{k\in K} \sum_{(i,j)\in N} ((DW + FC) \cdot t_{ij} + C_f \cdot f_{ij} + C_e \cdot f_{ij} + a_{ij} \cdot d_{ij} \cdot y_{ijk}) \cdot x_{ijk} \tag{B.7}$$

Based on Mirabi et al. (2010) and Caceres et al. (2014), the constraints are as follows:

$$\sum_{i\in N} \sum_{k\in K} x_{ijk} = 1 \qquad \forall\, j \in N_c \tag{B.8}$$

$$\sum_{i\in N} \sum_{j\in N_c} r_j \cdot x_{ijk} \leq Q \qquad \forall\, k \in K \tag{B.9}$$

$$\sum_{(i,j)\in N} d_{ij} \cdot x_{ijk} \leq D \qquad \forall\, k \in K \qquad \text{(B.10)}$$

$$U_{lk} - U_{jk} + |N| \cdot x_{ljk} \leq |N| - 1 \qquad \forall\, l, j \in N_c, k \in K \qquad \text{(B.11)}$$

$$\sum_{j\in N}(x_{ijk} - x_{jik}) = 0 \qquad \forall\, i \in N_c,\ k \in K \qquad \text{(B.12)}$$

$$\sum_{j\in N_c}\sum_{k\in K} x_{ijk} \leq 1 \qquad \forall\, i \in N_c \qquad \text{(B.13)}$$

$$\sum_{j\in N_c} x_{ijk} \leq 1 \qquad \forall\, i \in N_d,\ k \in K \qquad \text{(B.14)}$$

$$\sum_{i\in N_c} x_{ijk} \leq 1 \qquad \forall\, j \in N_d,\ k \in K \qquad \text{(B.15)}$$

$$\sum_{j\in N_c} x_{ijk} = \sum_{j\in N_c} x_{jik} \qquad \forall\, i \in N_d,\ k \in K \qquad \text{(B.16)}$$

$$-z_{ij} + \sum_{u\in N}(x_{iuk} + x_{ujk}) \leq 1 \qquad \forall\, i \in N_d, j \in N_c, k \in K \qquad \text{(B.17)}$$

$$\sum_{i\in N} y_{ijk} - \sum_{i\in N} y_{jik} = r_j \cdot \sum_{i\in N} x_{ijk} \qquad \forall\, j \in N_c,\ k \in K \qquad \text{(B.18)}$$

$$0 \leq r_j \cdot x_{ijk} \leq y_{ijk} \leq (Q - r_i) \cdot x_{ijk} \qquad \forall\, (i,j) \in N, i \neq j,\ k \in K \qquad \text{(B.19)}$$

$$x_{ijk} = 0 \qquad \forall\, (i,j) \in N_d,\ k \in K \qquad \text{(B.20)}$$

$$x_{ijk} \in \{0,1\} \qquad \forall\, i \in N_d,\ j \in N_c,\ k \in K \qquad \text{(B.21)}$$

$$z_{ij} \in \{0,1\} \qquad \forall\, i \in N_d,\ j \in N_c \qquad \text{(B.22)}$$

$$U_{lk} \geq 0 \qquad \forall\, l \in N_c,\ k \in K \qquad \text{(B.23)}$$

Equation B.8 assigns each customer to exactly one route. Equation B.9 limits the total demand that may be served by a vehicle. Equation B.10 defines the maximum distance per vehicle. Equation B.11 eliminates sub-tours. The flow conservation is introduced by Equation B.12. Equation B.13, B.14 and B.15 ensure that each route is completed once while Equation B.16 imposes that each route starts and ends at the same depot. Equation B.17 specifies that a customer can be assigned to a depot only if there is a route from that depot going through that customer. Equation B.18 states that the load in the vehicle arriving at customer j minus the demand of that customer equals the load in the vehicle leaving it after the service. Equation B.19 sets lower and upper bounds, which ensure that the load in the vehicle k leaving customer i is equal or greater than the demand of its next visit and the total demand serviced by the vehicle does not exceed its capacity, respectively. Equation B.20 avoids the creation of routes among depots. Finally, Equations B.21, B.22 and B.23 define variable domains.

4. Solving approach

The methodology proposed is based on the VNS metaheuristic. Besides being a popular metaheuristic in combinatorial as well as global optimization, it has been used in a wide range of research fields such as vehicle routing, scheduling, telecommunications, biology, and artificial intelligence. In essence, the metaheuristic proposes systematic changes of neighborhood to find a local minimum by intensifying the search, and to escape from the associated valley by diversifying. It relies on three facts: *i)* a local minimum with respect to one neighborhood structure is not necessarily so for another; *ii)* a global minimum is a local minimum with respect to all possible neighborhood structures; and *iii)* for many problems, local minima with respect to one or several neighborhoods are relatively close to each other. It was first proposed by Hansen and Mladenović (2014) and an extensive review may be found in Moreno-Vega and Melián (2008). Real-life applications requiring to make decisions tend to involve more than one criterion, besides a set of constraints that limits the solution space. In practice, these criteria may be conflicting. Here, all impacts are monetized relying on estimates proposed in the literature.

Our approach is summarized in Pseudo-code 1 and described next. The inputs are the problem instance to solve and the number of neighborhoods considered (K). It is usual in the literature to set K to two or three, and to design nested neighborhoods. First, an initial solution is generated and stored in *initSol* and *baseSol*. Then, the cost of all the impacts associated are computed.

bestSol will store the best solution found so far. At the beginning, it is a copy of *baseSol*. An outer loop is started, which will end when a given stopping criterion is met. It sets the current neighborhood to the first one. Inside, another loop builds and assesses new solutions. Within this loop, the base solution is initially shaken, generating a solution from the k-th neighborhood of *baseSol*. The resulting solution is stored in *newSol* and the corresponding total cost is computed (Pseudo-code 2). The variable *rpd* measures the relative percentage difference between the total cost of *newSol* and *baseSol*. If there is an improvement (i.e., $rpd < 0$), a local search is applied to *newSol*, the resulting solution is copied into *baseSol*, and the current neighborhood is set to the first. In addition, *bestSol* is updated if it applies. This constitutes a descendent phase aimed to find a local minimum. Otherwise, *newSol* is accepted and the current neighborhood is set to the first with a probability of $exp(-rpd)$. This acceptance criterion aims to avoid entrapment at local optimum and was first proposed in Hatami et al. (2015). It is based on the criterion of the simulated annealing metaheuristic but is simpler and has no parameters. In case of not accepting *newSol*, the next neighborhood is analyzed (i.e., k is set to $k + 1$). The inner loop is executed until the last neighborhood is explored (i.e., $k = K$). Finally, *bestSol* is returned.

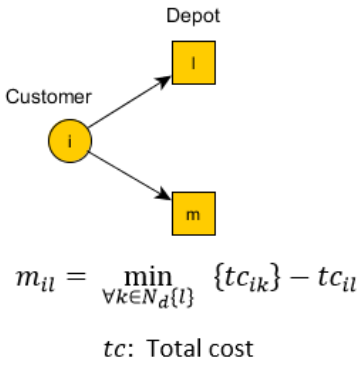
Algorithm 1 Approach for the MDVRP considering externalities.

```
1:  procedure MDVRP WITH SUSTAINABILITY INDICATORS (inputs, impactsParameters)
2:      initSol ← genInitSol (inputs) # generate solution based on the BR-CWS heuristic
3:      baseSol ← clone (initSol)
4:      computeTotalCost(baseSol, impactsParameters)
5:      bestSol ← clone (baseSol)
6:      while (stopping criterion is not met) do
7:          k ← 1
8:          while (k ≤ K) do
9:              newSol ← shake(baseSol, k) # destruction-construction stages
10:             computeTotalCost(newSol, impactsParameters)
11:             rpd ← (getTotalCost(newSol) - getTotalCost(baseSol))/getTotalCost(baseSol)· 100
12:             if (rpd < 0) then # newSol improves baseSol
13:                 newSol ← localSearch(newSol)
14:                 baseSol ← newSol
15:                 k ← 1
16:                 if (getTotalCost(newSol) - getTotalCost(bestSol) < 0) then
17:                     bestSol ← newSol
18:                 end if
19:             else
20:                 u ← generateU()
21:                 if (u < exp(−rpd)) then #acceptance criterion
22:                     baseSol ← newSol
23:                     k ← 1
24:                 else
25:                     k ← k + 1
26:                 end if
27:             end if
28:         end while
29:     end while
30:     bestSol ← localSearch(bestSol)
31:     return bestSol
32: end procedure
```

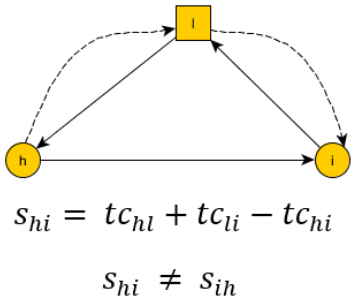
The generation of solutions for the MDVRP has two sequential and interrelated stages: *a)* the assignment of customers to depots, and *b)* the design of distribution routes for each depot. Both stages employ biased randomization techniques. These techniques allow the randomization of deterministic and iterative heuristics in order to obtain a relatively high number of promising solutions. For instance, the choice of one element or step from a list would be done by assigning, to each one, a probability according to a measure of preference instead of choosing the best considering only the next step. These techniques rely on the fact that the best step in the short term is not necessarily the best in the long term. The first stage of the generation of MDVRP solutions relies on a measure called "marginal savings" (Juan et al., 2015c) (Figure B.5a), which is computed for each pair depot-customer as follows: the distance between each depot and the

Algorithm 2 Function to monetize the impacts of a given solution.

```
1:  procedure COMPUTE TOTAL COST(MDVRPSol, impactsParameters)
2:      distance ← 0
3:      time ← 0
4:      emissions ← 0
5:      social ← 0
6:      for each (cvrpSol in MDVRPSol) do
7:          distance ← distance + getDistance(cvrpSol)
8:          time ← time + getTime(cvrpSol)
9:          for each (edge in cvrpSol) do
10:             emissions ← emissions + getDistance(edge)/getKPL(edge)
11:             social ← social + getDistance(edge) · getLoad(edge, cvrpSol)
12:         end for
13:     end for
14:     distanceCost ← distance· getDistUnitCost(impactsParameters)
15:     timeCost ← time· getTimeUnitCost(impactsParameters)
16:     emissionsCost ← emissions· getEmissionsUnitCost(impactsParameters)
17:     socialCost ← social· getSocialUnitCost(impactsParameters)
18:     totalCost ← distanceCost + timeCost + emissionsCost + socialCost
19:     return totalCost
20: end procedure
```



(A) Marginal savings used in the assignation stage.



(B) Rich savings employed in the BR-CWS heuristic.

FIGURE B.5: Representation of savings.

customer is obtained, and the difference of assigning the customer to the specific depot instead of the closest depot among the others is computed. A priority list of customers is created for each depot and sorted according to the marginal savings. Thus, high marginal savings are prioritized, since assigning the corresponding customer to another depot (which would be farther) could lead to a poor-quality solution. These lists are randomized assigning probabilities according to a Geometric distribution (see Figure B.6). Three different policies are iteratively applied to choose the depot to select the next customer to be assigned: *i)* all depots choose the first node in their list at a time, following consecutive turns (known as round robin criteria); *ii)* randomly; *iii)* the depot which the highest remaining capacity is selected. Thus, using biased randomization and different policies promotes the generation of different assignation-maps. The second stage is based on the randomized version of the CWS saving's heuristic (Juan et al., 2011a), which also depends on a Geometric distribution applied to the savings to iteratively choose one merge among all possible. However, the classical distance-based savings are replaced by "rich savings" including all costs (Figure B.5b).

The performance of each solution is computed ad the sum of the costs associated to the impacts considered: economic, environmental and social. The shaking procedure randomly selects a percentage p_k of customers to be assigned to a different depot. Afterwards, the procedure to construct solutions is applied to repair the solution. This movement is introduced to diversify the search. This search is guided by the base solution, since the shaking procedure applied at each

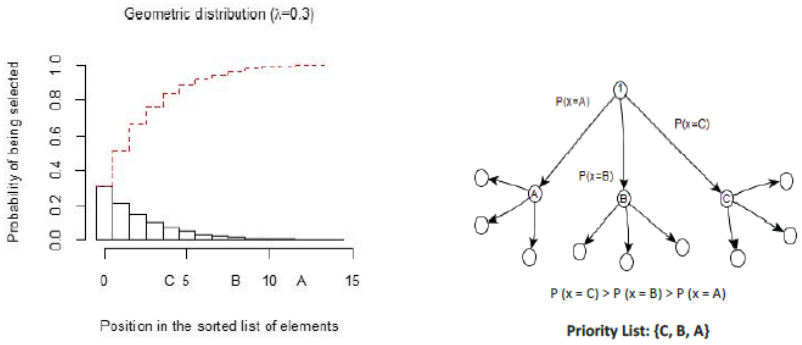


FIGURE B.6: Illustration of biased randomization techniques.

iteration works with that solution. It is set to the initial solution at the beginning and replaced by the new solution if the acceptance criterion is met. The stopping criterion is based on the number of iterations. Two local searches are used: the first is applied to solutions improving the current base solution and is based on the classical 2-opt operator defined for the CVRP (Lin, 1965), while the second is a routing extensive improving search described in Juan et al. (2011a), and applied only to the best solution found at the end. More details for specific procedures can be found in the references provided in this section.

5. Computational experiments

The algorithm proposed has been implemented in JAVA and run on a personal computer with 8 GB of RAM and an Intel Core i7 of 1.8 GHz. In order to test it, illustrate its use and the analysis of results that may be carried out, we employ 4 MDVRP benchmark instances (p10, p11, p12 and p13) called here instance 1, 2, 3 and 4, respectively. They have been extensively used (see Vidal et al., 2012; Escobar et al., 2014).

Each instance has been adapted as follows. Vehicles' efficiency parameters are based on a type of light duty vehicle used for freight distribution in urban zones. We have used the cost coefficients of Zhang et al. (2015) for CO2 emissions (C_e = 0.1 USD/L). Regarding the traveling time cost, it is defined by Koç et al. (2014) as the sum of a vehicle fixed cost (FC) and driver wage (DW), which are set to FC = 1.4 USD/h and DW = 6.3 USD/h, respectively. The traveling distance cost is based on the price of fuel (C_f = 1.1 USD/L) and the average miles per fuel liter (kpl_{ij} = 5.56 km/L $\forall$ $(i, j) \in N$). Delucchi and McCubbin (2010) propose an the interval $[1 \cdot 10^{-4}, 1.3 \cdot 10^{-3}]$ USD per kg-km for the coefficient a_{ij} to estimate the social cost. Without loss of generality, times are generated from distances using this formula $t_{ij} = \alpha \cdot d_{ij} + \varepsilon_{ij}$, where α is a constant based on an estimated speed (α^{-1} = 35 km/h) and ε_{ij} represents external factors that define the correlation between traveling time and distance. It is set to follow a truncated Normal distribution with a lower bound and mean equal to 0 and a standard deviation equal to 3.5, 2, and 0.5. These deviations are set in order to get a correlation around 0.5, 0.7 and 0.9, which may represent a high, medium and low congested zone, respectively. Thus, three scenarios are generated per instance. For example, for d_{ij}=10 km, t_{ij} would fall in the following intervals considering the different standard deviations and a probability of 95%: (0.59, 1.69), (0.64, 5.06), and (0.69, 8.42). The resulting instances are available from the authors upon request.

Each instance has been solved 10 times (employing a different seed for the random number generator) and only the best solutions are reported. 300,000 iterations are considered. The parameter fine-tuning is performed by using design of experiments and testing reasonable ranges. The parameters for the Geometric distributions related to the allocation and the routing process are randomly chosen in the intervals (0.5, 0.8) and (0.1, 0.2), respectively. The degree of shaking, which defines the neighborhoods, is set to 10%, 30% and 50% (for the first, second and third neighborhood, respectively).

The experimentation process consists on analyzing how the solution space changes according to the optimization criterion and how it influences the other indicators. Thus, five options are considered: optimization criterion is based on minimizing each component of the objective function or the sum of them. In a real-life application, the choice will depend on the particular interests of the decision-maker.

Validation

In order to validate our algorithm, the model described in Section 3 has been implemented in the GAMS language (Version 23.5.2), and the CPLEX solver (Version 12.2.0.0) has been used to obtain solutions. Three instances are created with 22, 30 and 40 nodes, respectively. They are derived from the instance p12, ensuring a feasible solution.

Table B.1 identifies the instance and the scenario, and contains information regarding the number of vehicles (V), customers (C), depots (D), and vehicles' capacity (Q). The next three columns refer to the solution found by CPLEX with the aim of minimizing the total cost. They represent this cost, the run time (limited to 36000 seconds, for each instance) and an upper bound for the gap between the solution found and the optimal (if the latter has not been found). The following two columns denote the total cost and the run time of our approach (OA), while the last column is the gap of the total cost between both approaches.

According to these results, the difference of computing times is significant, while the gaps provided by GAMS quickly increases with the number of nodes considered. It is important to note that the solver requires the number of vehicles that will depart from each depot, while the metaheuristic-based approach does not.

TABLE B.1: Comparison in terms of total cost between the CPLEX solver and the proposed metaheuristic.

Instance	Scenario	(V, C, D)	Q	CPLEX			OA		Gap (%)
				Total Cost	Run time (s)	Gap (%)	Total Cost	Run time (s)	
P1	High	(3, 20, 2)	60	305.13	3.44	0			
	Medium			466.36	5.20	0			
	Low			54.78	2.22	0			
P2	High	(2, 28, 2)	60	630.71	9800	18.87			
	Medium			831.95	9800	13.97			
	Low			908.41	9800	13.29			
P3	High	(2, 38, 2)	60	*	14314				
	Medium			1044.11	36000	21.74			
	Low			1103.66	36000	11.38			

* Out of memory

Moreover, our approach is compared against that described in Juan et al. (2015c), which only aims to minimize the total traveling distance. Table B.2 identifies the instance, describes their main characteristics and provides the distances of the aforementioned paper and of our approach, considering five seeds. The average computing times are 322 and Y, respectively. The mean gap among instances is -1.45%, which shows that the performance of both approaches is comparable.

TABLE B.2: Comparison in terms of distance between Juan et al. (2015c) and the proposed metaheuristic.

Instances	(V, C, D)	Q	JUAN15	OA	Gap (%)
p10	(8, 249, 4)	500	3646.67	3725.00	-2.10%
p11	(6, 249, 5)	500	3547.09	3604.6492	-1.60%
p12	(5, 80, 2)	60	1318.95	1331.54	-0.95
p13	(5, 80, 2)	60	1318.95	1334.4	-1.16

6. Discussion of results

This section analyzes the solutions found considering all the indicators or each one of them as optimization criterion. This comparison aims to determine a solution subspace representing an equilibrium between the economic, environmental, and social dimensions.

Table B.3 shows the total cost of the best solutions found according to the objective pursued for each instance and scenario. As described before, the total cost is computed as the sum of the costs associated to traveling distance, traveling time and CO2 emissions, and the social cost. As expected, the best solution in terms of total cost is the solution that seeks to minimize the total cost. In instance 1 and 2, the solution minimizing the traveling time matches the solution minimizing

the total cost, which means that these objectives converge to the same solution. Similarly, the same solution ensures the minimum traveling distance cost and emissions cost (this is due to the way in which the emissions are estimated). Obviously, the total cost is higher in the zones where the traveling time and the traveling distance have a low correlation (i.e., in congested zones, based on the description of the scenarios). The solution with the minimum social cost is the most expensive, because the other costs are significantly increased.

TABLE B.3: Total cost by scenario, instance and optimization criterion.

		Scenarios			
		Low	Medium	High	
Instance	Objective	Total Cost	Total Cost	Total Cost	Run Time (s)
Instance 1	Total Cost	7597.4	5669.1	3763.3	1665.8
	Distance	8601.8	6054.1	3763.3	1572.9
	Time	7597.4	5770.8	3867.3	1688.3
	CO2 Emissions	8601.8	6054.1	3763.3	1572.9
	Social cost	8686.3	6393.7	3977.7	1485.0
Instance2	Total Cost	7625.8	5979.4	3645.0	1368.4
	Distance	9087.9	6392.3	3645.0	1625.2
	Time	7625.8	6060.9	3741.2	1603.2
	CO2 Emissions	9087.9	6392.3	3645.0	1625.2
	Social cost	9096.5	6651.7	3898.9	1130.3
Instance 3	Total Cost	2475.6	1913.3	1197.9	200.2
	Distance	2949.0	1944.5	1199.6	190.8
	Time	2475.6	1949.0	1197.9	198.5
	CO2 Emissions	2949.0	1944.5	1199.6	190.8
	Social cost	2979.8	1999.8	1241.6	186.0
Instance 4	Total Cost	2757.8	1904.3	1217.0	185.9
	Distance	2871.1	1913.3	1217.0	187.3
	Time	2813.3	1927.9	1222.7	189.3
	CO2 Emissions	2871.1	1913.3	1217.0	183.9
	Social cost	3144.2	2103.1	1385.6	182.8

Regarding the social cost, it is important to determine the customer sequence and the direction of the route. Figure B.7 illustrates and quantifies their effect on the total cost of a given route. Accordingly, high-quality solutions visits first the customers with higher demands, minimizing the amount of freight transported over long stretch of roads. On the other hand, the scenario influences the total cost. Table B.3 suggests that the gap between solutions with minimum total cost and minimum social cost is higher in congested zones. This happens because minimizing the social cost involves reducing the traveling distance, which leads to optimize also the traveling time if there is a high correlation between time and distance.

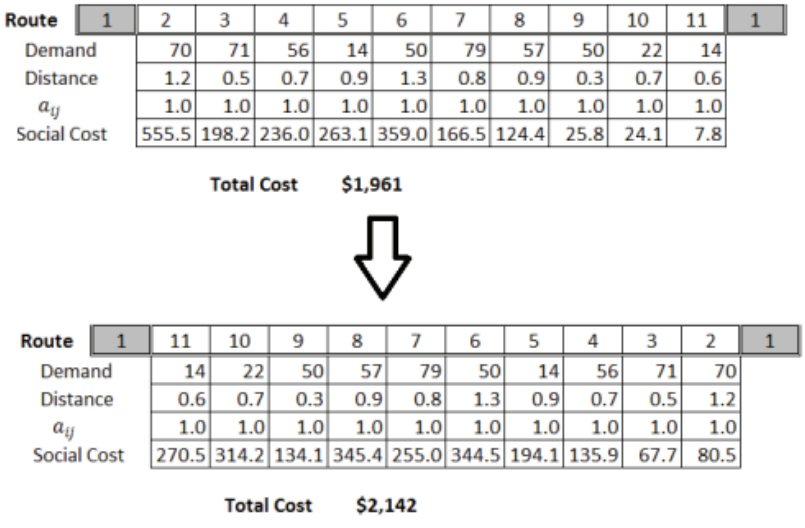
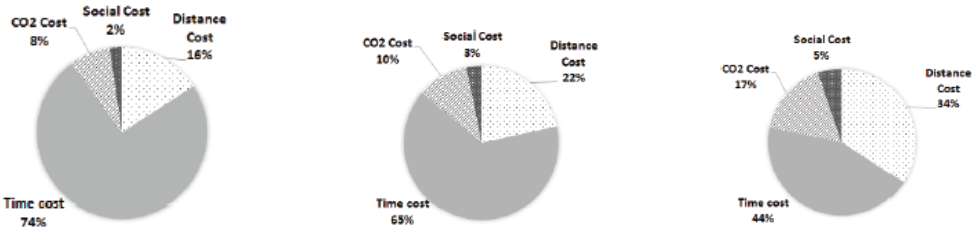


FIGURE B.7: Effect of the customer sequence and the direction for a given route.

Figures B.8 and B.9 provide information regarding the behavior of solutions for each scenario. The first represents the average weight of each cost component per scenario considering the four instances. It can be observed that traveling time represents the main cost and its magnitude is the most sensitive to the scenario. Figure B.9 shows the ranges of total cost and its components per scenario for instance 1. In this case, the time cost increases at a higher rate than the distance cost, which causes differences among scenarios.



(A) Low Scenario (B) Medium Scenario (C) High Scenario

FIGURE B.8: Average weight of each component in the total cost by scenario considering all instances.

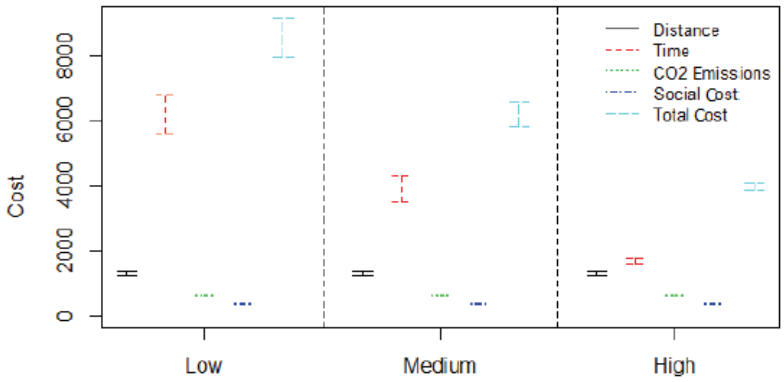


FIGURE B.9: Total cost and its component per scenario for instance 1.

Table B.4 shows the cost of each indicator when the main objective is to minimize the total cost for all instances and scenarios. The gaps reflect the difference between the solution with minimum total cost and the best solution for each indicator. For example, the solution with the minimum total cost for instance 1 in the low scenario has a social cost 9.49% higher than the best solution found when the objective is to minimize the social cost. This table demonstrates that the solution with the minimum total cost does not tend to be the best when applying another optimization criterion.

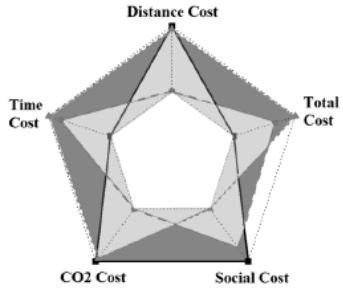
Figure B.10 displays radar plots for instances 1 and 4, and the scenarios low and high using the best solutions found for each indicator and the total cost. These plots identify the desirable and sustainability regions. The desirable region may be used to define an upper bound (or maximum allowable cost for each measure) and a lower bound (i.e., the white regular pentagon), which represents the ideal solution. There is no guarantee that a feasible solution exists that falls in the sustainability region. However, if one is found, it can be argued that that solution achieves a suitable balance between at least two measures. In our case, the sustainability region is a narrower area for the scenario with less nodes and a high correlation between traveling time and distance.

7. Conclusions and future research

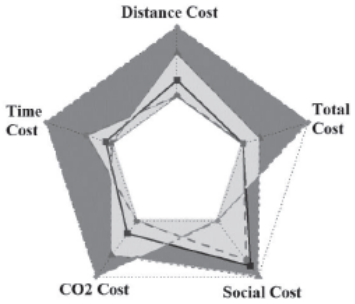
The increasing concern for the environment and the population welfare leads policy-makers to promote a sustainable growth. In this context, urban transportation plays an essential role because of the relevance of its negative impacts. Despite the extensive literature on routing problems and a

TABLE B.4: Comparison among solutions for each instance and scenario.

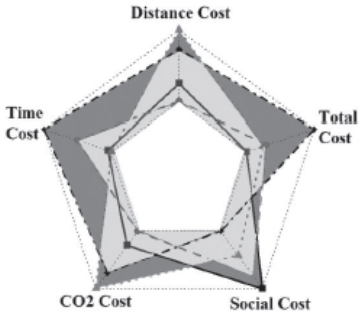
Objective: Minimizing Total cost										
Scenario	Instance		Traveling distance		Traveling time		CO2 emissions		Social cost	
		(V, C, D)	Cost	Gap (%)	Cost	Gap (%)	Cost	Gap (%)	Cost	Gap (%)
Low	1	(8,249,4)	1386.35	-12.88%	5136.15	0.00%	672.10	-12.88%	402.78	-20.37%
	2	(6,249,5)	1376.06	-14.96%	5192.52	0.00%	667.11	-14.96%	390.11	-21.38%
	3	(5,80,2)	560.84	-23.09%	1624.93	0.00%	271.89	-23.09%	17.90	-31.55%
	4	(5,80,2)	460.19	-5.66%	2059.82	-1.16%	223.10	-5.66%	14.69	-28.31%
		Average	945.86	-14.15%	3503.36	-0.29%	458.55	-14.15%	206.37	-25.40%
Medium	1	(8,249,4)	1328.31	-9.07%	3325.69	-2.38%	643.96	-9.07%	371.14	-13.58%
	2	(6,249,5)	1312.04	-10.81%	3671.20	-2.96%	636.08	-10.81%	360.05	-14.82%
	3	(5,80,2)	434.16	-0.65%	1254.84	-12.01%	210.48	-0.65%	13.78	-11.05%
	4	(5,80,2)	442.88	-1.97%	1232.64	-0.26%	214.71	-1.97%	14.09	-25.23%
		Average	879.35	-5.63%	2371.09	-4.40%	426.31	-5.63%	189.77	-16.17%
High	1	(8,249,4)	1204.67	0.00%	1626.95	-4.72%	584.02	0.00%	347.65	-7.74%
	2	(6,249,5)	1177.17	-0.59%	1563.70	-3.79%	570.69	-0.59%	333.45	-8.03%
	3	(5,80,2)	435.35	-0.92%	538.16	0.00%	211.06	-0.92%	13.33	-8.06%
	4	(5,80,2)	434.16	0.00%	558.53	-0.48%	210.48	0.00%	13.78	-23.54%
		Average	812.84	-0.38%	1071.84	-2.25%	394.06	-0.38%	177.05	-11.84%



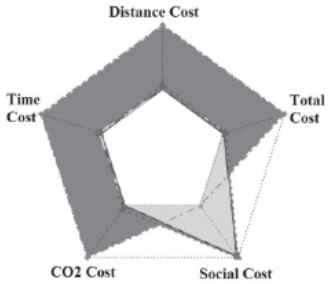
(A) Instance 1: Low Scenario



(B) Instance 1: High Scenario



(C) Instance 4: Low Scenario



(D) Instance 4: High Scenario

FIGURE B.10: Solution spaces for decision-making.

recent trend to include environmental issues, there is a lack of works combining the three axis of sustainability and studying their relations.

This work aims to reduce the existing gap by presenting a new definition of the multi-depot vehicle routing problem employing sustainability indicators. Adopting this perspective can be seen as a market strategy. Estimates from the literature are considered to quantify and monetize the economic, environmental and social impacts. While the first are based on traveling distances and times, the second and the third rely on carbon emissions and risk of accidents, respectively. We develop a solving approach based on the variable neighborhood search metaheuristic, which integrates a biased randomization version of the classical Clarke and Wright's savings heuristic. This methodology is able to report high-quality solutions in small amounts of time and enables decision-makers to assess solutions under particular interests regarding the impacts considered. A set of computational experiments are carried out in order to test our algorithm and illustrate its use, and present visualization techniques that may help the decision-makers to better understand a few promising options and their implications.

Several lines of future research stem from this work. First, the introduction of sustainability indicators in richer vehicle routing problems can be explored, for instance, considering electric vehicles, which incorporates more restrictions, and/or a heterogeneous fleet of vehicles. Similarly, uncertainty of real-life applications could be taken into account by modeling involved inputs such as traveling times or demands as stochastic variables. Another interesting line would be to implement our methodology for a case study.

Acknowledgments

This work has been partially supported by the Spanish Ministry of Economy and Competitiveness and FEDER (TRA2013-48180-C3-P, TRA2015-71883-REDT), and the Erasmus+ programme (2016-1-ES01-KA108-023465). Likewise, we want to thank the support of the UOC and UPNA doctoral programme.

References

Bektaş, T. and G. Laporte (2011). "The pollution-routing problem". In: *Transportation Research Part B: Methodological* 45.8, pp. 1232–1250.

Caceres, J., P. Arias, D. Guimarans, D. Riera, and A. A. Juan (2014). "Rich vehicle routing problem: a survey". In: *ACM Computing Surveys* 47.2. ISSN: 0360-0300.

Chen, L.-W., P. Sharma, and Y.-C. Tseng (2013). "Dynamic traffic control with fairness and throughput optimization using vehicular communications". In: *IEEE Journal on Selected Areas in Communications* 31.9, pp. 504–512.

Chibeles-Martins, N., T. Pinto-Varela, A. P. Barbosa-Póvoa, and A. Q. Novais (2016). "A multi-objective meta-heuristic approach for the design and planning of green supply chains". In: *Expert Systems with Applications* 47, pp. 71–84.

Clarke, G. and J. Wright (1964). "Scheduling of vehicles from a central depot to a number of delivering points". In: *Operations Research* 12, pp. 568–581.

Delucchi, M. A. and D. R. McCubbin (2010). *External costs of transport in the US*. Tech. rep.

Demir, E., T. Van Woensel, and T. de Kok (2014). "Multidepot distribution planning at logistics service provider Nabuurs BV". In: *Interfaces* 44.6, pp. 591–604.

European Commission (2015). *EU Transport in figures*. Statistical pocketbook.

Eurostat (2015). *Sustainable development in the European Union: 2015 monitoring report of the EU sustainable development strategy*. Publications office of the European Union.

Faulin, J., A. Juan, F. Lera, and S. Grasman (2011). "Solving the capacitated vehicle routing problem with environmental criteria based on real estimations in road transportation: a case study". In: *Procedia-Social and Behavioral Sciences* 20, pp. 323–334.

Hansen, P. and N. Mladenović (2014). "Variable neighborhood search". In: *Search methodologies*. Springer, pp. 313–337.

Hunt, J. D. and K. J. Stefan (2007). "Tour-based microsimulation of urban commercial movements". In: *Transportation Research Part B: Methodological* 41.9, pp. 981–1013.

Jabbarpour, M. R., R. Md. Noor, and R. H. Khokhar (2015). "Green vehicle traffic routing system using ant-based algorithm". In: *Journal of Network and Computer Applications* 58, pp. 294–308.

Juan, A. A., J. Faulin, R. Ruiz, B. Barrios, and S. Caballe (2010). "The SR-GCWS hybrid algorithm for solving the capacitated vehicle routing problem". In: *Applied Soft Computing* 10.1, pp. 215–224.

Juan, A. A., J. Faulin, J. Jorba, D. Riera, D. Masip, and B. Barrios (2011a). "On the use of Monte Carlo simulation, cache and splitting techniques to improve the Clarke and Wright savings heuristics". In: *Journal of the Operational Research Society* 62, pp. 1085–1097.

Juan, A. A., J. Faulin, E. Pérez-Bernabeu, and N. Jozefowiez (2014e). "Horizontal cooperation in vehicle routing problems with backhauling and environmental criteria". In: *Procedia-Social and Behavioral Sciences* 111, pp. 1133–1141.

Juan, A. A., I. Pascual, D. Guimarans, and B. Barrios (2015c). "Combining biased randomization with iterated local search for solving the multidepot vehicle routing problem". In: *International Transactions in Operational Research* 22.4, pp. 647–667.

Kadziński, M., T. Tervonen, M. K. Tomczyk, and R. Dekker (2017). "Evaluation of multi-objective optimization approaches for solving green supply chain design problems". In: *Omega* 68, pp. 168–184.

Koç, Ç., T. Bektaş, O. Jabali, and G. Laporte (2014). "The fleet size and mix pollution-routing problem". In: *Transportation Research Part B: Methodological* 70, pp. 239–254.

Kuo, Y. (2010). "Using simulated annealing to minimize fuel consumption for the time-dependent vehicle routing problem". In: *Computers & Industrial Engineering* 59.1, pp. 157–165.

Lin, S. (1965). "Computer solutions of the traveling salesman problem". In: *Bell System Technical Journal* 44.10, pp. 2245–2269.

Liu, W.-Y., C.-C. Lin, C.-R. Chiu, Y.-S. Tsao, and Q. Wang (2014). "Minimizing the carbon footprint for the time-dependent heterogeneous-fleet vehicle routing problem with alternative paths". In: *Sustainability* 6.7, pp. 4658–4684.

Lourenço, H.R., O.C. Martin, and T. Stützle (2010). "Iterated local search: framework and applications". In: *Handbook of Metaheuristics*. Ed. by M. Gendreau and J.-Y. Potvin. Vol. 146. International Series in Operations Research & Management Science. Springer US, pp. 363–397.

McKinnon, A., M. Browne, A. Whiteing, and M. Piecyk (2015). *Green logistics: improving the environmental sustainability of logistics*. Kogan Page Publishers.

Mirabi, M., S.M.T. Fatemi-Ghomi, and F. Jolai (2010). "Efficient stochastic hybrid heuristics for the multi-depot vehicle routing problem". In: *Robotics and Computer-Integrated Manufacturing* 26.6, pp. 564–569.

Muñuzuri, J., P. Cortés, L. Onieva, and J. Guadix (2010). "Modelling peak-hour urban freight movements with limited data availability". In: *Computers & Industrial Engineering* 59.1, pp. 34–44.

Niknamfar, A. H. and S. T. A. Niaki (2016). "Fair profit contract for a carrier collaboration framework in a green hub network under soft time-windows: Dual lexicographic max–min approach". In: *Transportation Research Part E: Logistics and Transportation Review* 91, pp. 129–151.

Ranaiefar, F. and R. Amelia (2011). "Freight-transportation externalities". In: *Logistics Operations and Management*. Ed. by R. Z. Farahani, S. Rezapour, and L. Kardar. Elsevier, pp. 333 –358. ISBN: 978-0-12-385202-1.

Reyes, L., L. Calvet, A. A. Juan, and J. Faulin (submitted). "Sustainable urban freight transport considering multiple capacitated depots". In: *A*.

Ruan, M., J. J. Lin, and K. Kawamura (2012). "Modeling urban commercial vehicle daily tour chaining". In: *Transportation Research Part E: Logistics and Transportation Review* 48.6, pp. 1169–1184.

Santos, G., H. Behrendt, L. Maconi, T. Shirvani, and A. Teytelboym (2010). "Part I: Externalities and economic policies in road transport". In: *Research in Transportation Economics* 28.1, pp. 2–45.

Scheuer, S. (2005). *EU environmental policy handbook. A critical analysis of EU environmental legislation: making it accessible to environmentalists and decision makers*. European Environmental Bureau.

Talbi, E.-G. (2009). *Metaheuristics: From design to implementation*. New Jersey: John Wiley & Sons.

Teo, J. S. E., E. Taniguchi, and A. G. Qureshi (2012). "Evaluating city logistics measure in e-commerce with multiagent systems". In: *Procedia-Social and Behavioral Sciences* 39, pp. 349–359.

Ubeda, S., F.J. Arcelus, and J. Faulin (2011). "Green logistics at Eroski: A case study". In: *International Journal of Production Economics* 131.1, pp. 44–51.

Wang, X., H. Kopfer, and M. Gendreau (2014b). "Operational transportation planning of freight forwarding companies in horizontal coalitions". In: *European Journal of Operational Research* 237.3, pp. 1133–1141.

Wang, X., T. Fan, W. Li, R. Yu, D. Bullock, B. Wu, and P. Tremont (2016). "Speed variation during peak and off-peak hours on urban arterials in Shanghai". In: *Transportation Research Part C: Emerging Technologies* 67, pp. 84–94.

Xiao, Y. and A. Konak (2015). "Green vehicle routing problem with time-varying traffic congestion". In: *Proceedings of the 14th INFORMS Computing Society Conference*, pp. 134–148.

Xie, K., X. Wang, H. Huang, and X. Chen (2013). "Corridor-level signalized intersection safety analysis in Shanghai, China using Bayesian hierarchical models". In: *Accident Analysis & Prevention* 50, pp. 25–33.

Zhang, J., Y. Zhao, W. Xue, and J. Li (2015). "Vehicle routing problem with fuel consumption and carbon emission". In: *International Journal of Production Economics* 170, pp. 234–242.

Zhang, S., C. K. M. Lee, K. Wu, and K. L. Choy (2016). "Multi-objective optimization for sustainable supply chain network design considering multiple distribution channels". In: *Expert Systems with Applications* 65, pp. 87–99.

Appendix C

Selected journal papers indexed in Elsevier-Scopus

C.1 Educational data mining and learning analytics: Differences, similarities, and time evolution

Laura Calvet, Angel A. Juan

Department of Computer Science, Open University of Catalonia, IN3, 08018 Barcelona, Spain
e-mail: {lcalvetl, ajuanp}@uoc.edu

Abstract

Technological progress in recent decades has enabled people to learn in different ways. Universities now have more educational models to choose from, i.e., b-learning and e-learning. Despite the increasing opportunities for students and instructors, online learning also brings challenges due to the absence of direct human contact. Online environments allow the generation of large amounts of data related to learning/teaching processes, which offers the possibility of extracting valuable information that may be employed to improve students' performance. In this paper, we aim to review the similarities and differences between Educational Data Mining and Learning Analytics, two relatively new and increasingly popular fields of research concerned with the collection, analysis, and interpretation of educational data. Their origins, goals, differences, similarities, time evolution, and challenges are addressed, as are their relationship with Big Data and MOOCs.

Keywords: Online Learning, Educational Data Mining, Learning Analytics, Big Data.

1. Introduction

In the traditional educational model, instructors have the principal role in the learning process. Students are assumed to have basic knowledge and skills, while instructors are expected to share their knowledge and experience. Learning is tested by means of proctored exams and homework. Before the Internet era, there were several types of distance-education models based on TV programmes, manuals or recorded audios/videos. Typically, instructors were available to solve doubts by phone or mail. Although they allowed learning from home and presented a flexible timetable, the lack of interactivity hindered the learning process.

The Internet has dramatically changed the system, since most institutions have become interested in providing online courses. Besides the fact that they do not require large investments, these courses are not restricted to a specific geographical location or timetable, which increases the number of potential students. As a result, universities dedicated only to online education have emerged and traditional universities have expanded their offer with b-learning (hybrid classroom and online learning) and e-learning (pure online learning) courses.

As Daradoumis et al. (2010b) state, e-learning has many more positive aspects: (a) it favours interactive communication among students, and between students and instructors; (b) it promotes continuous evaluation based on tests, and individual and collaborative activities; (c) it contributes to the development of technical skills; and (d) it helps to reduce the gap between theory and practice (e.g., Marquès et al., 2013). The role of the instructor is to design, organize and support

learning experiences. While in the traditional model all students listen to the same lectures and complete the same homework in the same sequence and at the same pace (Bienkowski et al., 2012), this model promotes a more personalized learning process, in which the student has an active role. However, e-learning courses also present higher dropout rates due to the fact that distance education may create a sense of isolation in students, which can feel disconnected from the other students, the instructors and the university (Juan et al., 2009b).

E-learning courses may be provided through Learning Management Systems (LMS) such as Moodle, Sakai and ILIAS, or Learning Platforms such as Knewton and DreamBox. A characteristic of these courses is the vast amount of data that can be collected. In addition to student's background and performance data, each action carried out (reading files, participating in forums, sending messages, or visiting recommended links, for example) leaves a digital fingerprint.

There are two fields of research devoted to analyzing this data: Educational Data Mining (EDM) and Learning Analytics (LA). Their overwhelming popularity is almost certainly due to several factors: (a) there is interest in employing a data-driven approach to make better decisions, as it is usual in business intelligence or analytics (Daradoumis et al., 2010a); (b) there are powerful statistical, machine-learning and data-mining methods and techniques to search for patterns in data and construct predictive models or decision rules that can be easily adapted to educational data; (c) generating data is relatively easy, and current computer capacity allows its storage and processing; (d) because of the financial crisis and fierce competition, universities are under pressure to reduce costs and increase income by exploiting the growing educational demands from developing countries, reducing dropout rates and improving course quality.

The main goal of both EDM and LA is to extract information from educational data to support education-related decision making. Information may be oriented towards several stakeholders (Daradoumis et al., 2010b). Instructors may get more objective feedback to evaluate both the structure of their courses and the effectiveness of the learning process. Monitoring the students' learning process may help to rapidly spot those having difficulties in following the course, and units that generate more confusion. It can be a complex and time-consuming task without the appropriate tools (Juan et al., 2009a). Students may receive recommendations about resources according to their performance, goals and motivations, may graphically analyze the outputs of their learning process, compare them with those of the rest of the class, and observe the performance and contributions related to collaborative activities. Managers may use information to design a better allocation of human and material resources to improve the overall quality of their academic offer. Finally, researchers may test and adapt their theories based on educational data.

Some initial similarities and differences between EDM and LA will be discussed in this paper. From a general perspective, it can be argued that EDM focuses more on techniques and methodologies, while LA deals more with applications. However, as we will see, these differences seem to be less and less noticeable as both fields evolve over time. In addition, the most significant barriers to EDM and LA applications in educational environments and a few hot research topics will be mentioned. Accordingly, the contributions of this work are: (a) to analyze the origins and particularities of these fields of research; (b) to provide an overview of the associated literature; (c) to examine how both knowledge areas have evolved in recent years and to discuss their possible convergence; and (d) to present some of the challenges and new trends, including those related with Big Data and MOOCs.

The rest of this paper is organized as follows: Section 2 and 3 offer an introduction to EDM and LA, respectively; Section 4 reviews some common methods, and Section 5 points out the main similarities and differences between these concepts; Section 6 identifies the principal issues that still need to be addressed and explores the latest lines of research; finally, general conclusions are drawn in Section 7.

2. Educational Data Mining

EDM develops and adapts statistical, machine-learning and data-mining methods to study educational data generated basically by students and instructors. Their application may help to analyze student learning processes considering their interaction with the environment (Baker and Altheimer, 2012). Initially, some workshops were held at conferences on Artificial Intelligence in Education and Intelligent Tutoring Systems. The first International Conference on EDM (Baker et al., 2008) was held in 2008 in Montreal. It has been held every year since then. The most popular societies are the International Educational Data Mining Society

(http://www.educationaldatamining.org/) created in 2011, and the IEEE Task Force of Educational Data Mining (http://datamining.it.uts.edu.au/edd/) formed in 2012. The related literature is extensive and varied. A commonly cited report is presented in Bienkowski et al. (2012), who introduce EDM and LA and also their bases, implementation challenges and application areas. Special consideration is given to Adaptive Learning Systems, which adapt learning experiences based on model predictions. As far as we are concerned, there are three books that detail applications and methods: Romero and Ventura (2006), Romero et al. (2010), and Peña-Ayala (2013). Romero and Ventura (2010) present a survey with more than 300 references.

Applications of EDM methods comprise several steps (Figure C.1). Initially, a design is planned, i.e., the main aim of the study and the required data are identified. Afterwards, the data is extracted from the appropriate educational environment. Frequently, data will need to be pre-processed, since it may come from several sources or have different formats and levels of hierarchy. Models or patterns are obtained from applying EDM methods, which have to be interpreted. If the conclusions suggest applying changes to the teaching/learning process or are not conclusive (because the problem has not been adequately addressed, the raw data are small or not suitable, or the selected methods are not powerful enough), the analysis is performed again after modifying the teaching/learning process or the study design.

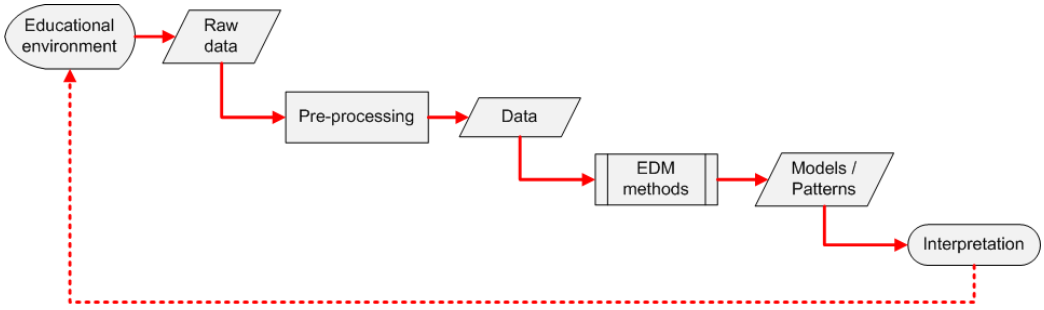


FIGURE C.1: Overview of how EDM methods are applied.

There are increasing numbers of EDM applications. According to Baker and Altheimer (2012), they can be grouped into the following four categories:

1. Student modelling: student data (including knowledge, motivations, etc.) and EDM techniques may be used to design a customized learning process by modelling differences between students.

2. Modelling of the knowledge structure of the domain: methods combining psychometric modelling frameworks with space-searching algorithms are created for discovering data-based domain models.

3. Pedagogical support: efficient educational support may be identified.

4. Scientific research: applications may help to develop and test educational scientific theories and to formulate new hypotheses.

Specific applications are described in Romero and Ventura (2013): predicting student performance, scientific inquiry, providing feedback for supporting instructors, personalizing/recommending to students, creating alerts for stakeholders (in real time in the event of undesirable student behaviours), student modelling (developing and tuning cognitive models of students, which represent their skills and declarative knowledge), domain modelling, student grouping/profiling, constructing courseware, planning and scheduling (related to courses, student scheduling, resource allocation, etc.), and parameter estimation.

A huge variety of tools have been designed and implemented to deploy EDM methods. However, most of these tools include a limited subset of the existing methods, are not publicly available or have been tested only in case studies. García et al. (2011) provide a list of them and point out that they are usually too complex for instructors without a background in data mining. Besides being easy to interpret and use, tools should be fast, especially in monitoring learning processes, where risk of dropouts and group internal conflicts may be better addressed if instructors are alerted before they occur (Juan et al., 2009a).

While most LMSs incorporate their own tools to automatically generate customizable statistics reports of course development, these are often quite basic. For instance, Moodle (https://moodle.org/) allows several types of report to be generated: (a) logs for selected activities, students, items and periods of time; (b) live logs, which include recent activity; (c) activity reports, presenting the numbers of views of each activity in a course; (d) course participation, analyzing the actions of selected students for a given period and activity; and (e) data on activity completion. Blackboard (http://es.blackboard.com/sites/international/globalmaster/) also offers several types of report, e.g., (a) user activity overview, which displays overall system and course activity for all students; (b) user statistics, consisting of the average number of students and other users per month and per day; (c) user activity in forums; and (d) user activity in groups. Another interesting tool that can be easily employed is Google Analytics (Figure C.2). It can provide information about the number of visits, pages visited, the average duration of each visit, demographics, etc. Regarding monitoring student activity and performance, Lera-López et al. (2009) review the tools provided by Sakai, WebCT/Blackboard and Moodle.

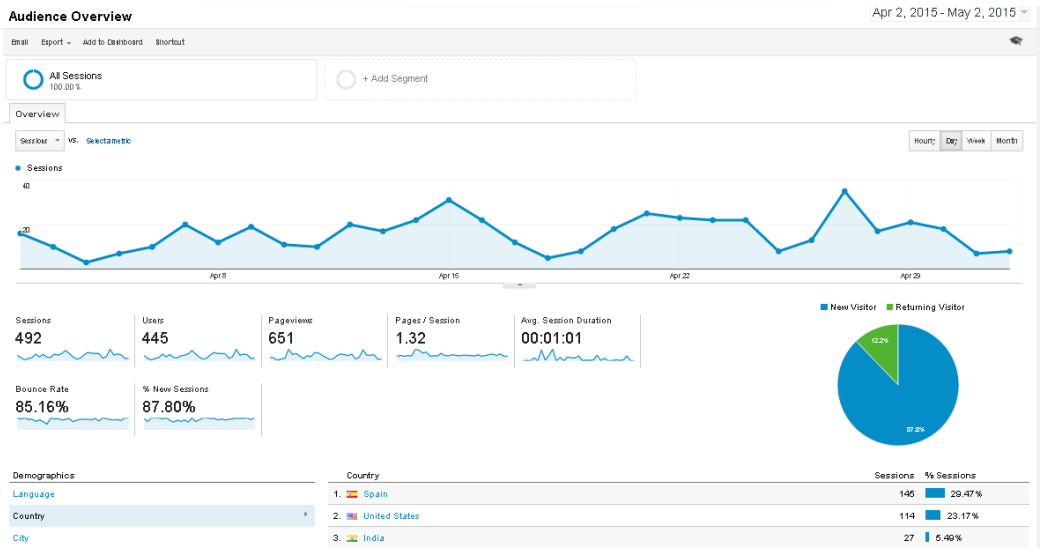


FIGURE C.2: Example of a Google Analytics report.

3. Learning Analytics

According to the call of the First International Conference on Learning Analytics and Knowledge (LAK) (https://tekri.athabascau.ca/analytics/), LA can be defined as the measurement, collection, analysis and reporting of data about learners and their contexts, for the purposes of understanding and optimising learning and the environments in which it occurs. The first International Conference on LAK (Long et al., 2011) was held in 2011, also in Canada. It has been held annually since then. The most active professional society was founded in the same year: the Society for Learning Analytics Research (SoLAR) (http://www.solaresearch.org). The book by Larusson and White (2014) is one of the main LA contributions to the literature. It includes the latest theories, findings, strategies, tools and case studies, and focuses on the following uses: (a) how to enhance student and faculty performance; (b) how to improve student understanding of course material; (c) how to assess and attend to the needs of struggling learners; (d) how to improve accuracy in grading; (e) how to allow instructors to assess and develop their own strengths; and (f) how to encourage more efficient use of resources at the institutional level.

The basic steps to test a learning/teaching process-related hypothesis are the same as those explained for EDM: an iterative process in which data is extracted from an educational environment and pre-processed before applying computational / quantitative methods in order to support stakeholders (instructors, course managers, etc.) when making decisions.

4. Common methods in EDM and LA

Most methods applicable to educational data are employed in both EDM and LA. The most popular are related to prediction, clustering and relationship mining. However, there are many more that cover a wide range of applications. The methods, their descriptions and a few examples are shown in Table C.1.

TABLE C.1: Common EDM-LA methods. Source: adapted from Romero and Ventura (2013).

Method	Goal/description	Key applications	Example
Prediction	To infer a target variable from some combination of other variables. Classification, regression and density estimation are types of prediction methods.	Predicting student performance and detecting student behaviours.	Yadav and Pal (2012)
Clustering	To identify groups of similar observations.	Grouping similar materials or students based on their learning and interaction patterns.	Antonenko et al. (2012)
Relationship mining	To study relationships among variables and to encode rules. Association rule mining, sequential pattern mining, correlation mining and causal data mining are the main types.	Identifying relationships in learner behaviour patterns and diagnosing student difficulties.	Kinnebrew and Biswas (2012)
Distillation of data for human judgment	To represent data in intelligible ways using summarization, visualization and interactive interfaces.	Helping instructors to visualize and analyze the ongoing activities of the students and the use of information.	Baker et al. (2006)
Discovery with models	To employ a previously validated model of a phenomenon as a component in another analysis.	Identification of relationships among student behaviours and characteristics or contextual variables. Integration of psychometric modelling frameworks into machine-learning models.	Jeong and Biswas (2008)
Outlier detection	To point out significantly different individuals.	Detection of students with difficulties or irregular learning processes.	Ueno (2004)
Social network analysis	To analyze the social relationships between entities in networked information.	Interpretation of the structure and relations in collaborative activities and interactions with communication tools.	Palazuelos et al. (2013)
Process mining	To obtain knowledge of the process from event logs.	Reflecting student behaviour in terms of its examination traces, consisting of a sequence of course, grade and timestamp.	Trčka et al. (2010)
Text mining	To extract high-quality information from text.	Analysing the contents of forums, chats, web pages and documents.	Tane et al. (2004)
Knowledge tracing	To estimate student mastery of skills, employing both a cognitive model that maps a problem-solving item to the skills required, and logs of students' correct and incorrect answers as evidence of their knowledge on a particular skill.	Monitoring student knowledge over time.	Lee and Brunskill (2012)
Nonnegative matrix factorization	To define a matrix M of positive numbers with student test outcome data that may be decomposed into two matrices: Q, which represents a matrix of items, and S, which represents student mastery of skills.	Assessment of student skills.	Desmarais (2012)

Baker and Trietsch (2009) study the proportion of works employing each group of methods during the period from 1995 to 2005 (using data extracted from Romero and Ventura (2007)) and from 2008 to 2009 (using data from Baker et al. (2008), and Barnes et al. (2009)). Papers from the first period mainly involved relationship mining methods (43%) or prediction methods (28%). Human judgment or exploratory data analysis (17%) and clustering (15%) were also popular. In contrast, relationship mining in the next period slipped to 5th place (9%), while prediction methods reached 1st place (42%, papers from 2008 only). The proportion using human judgment and clustering methods did not change considerably (12% and 15%, respectively). Discovery with models gained representation (19%), since no paper from the first period used this method. Also worthy of note is the importance of item response theory, Bayesian nets and Markov decision processes (28%).

5. Similarities and differences between EDM and LA

The overlap between both fields of research is certainly considerable. Even so, some differences are highlighted in the literature. EDM and LA have the same goal: improving education quality by analysing huge amounts of data to extract useful information for stakeholders. Representative companies in other sectors, such as industry, finance or healthcare, have already introduced statistical, machine-learning and data-mining techniques to achieve better performance through decisions based on historical data. The popularity of these fields of research has been growing since the early 2010s (Figure C.3), although EDM research started a few years beforehand. It is expected that these fields will continue to expand (Johnson et al., 2012), due to the potential benefits (for students, instructors, administrators, researchers and society in general) and the relevance of current research based on Big Data.

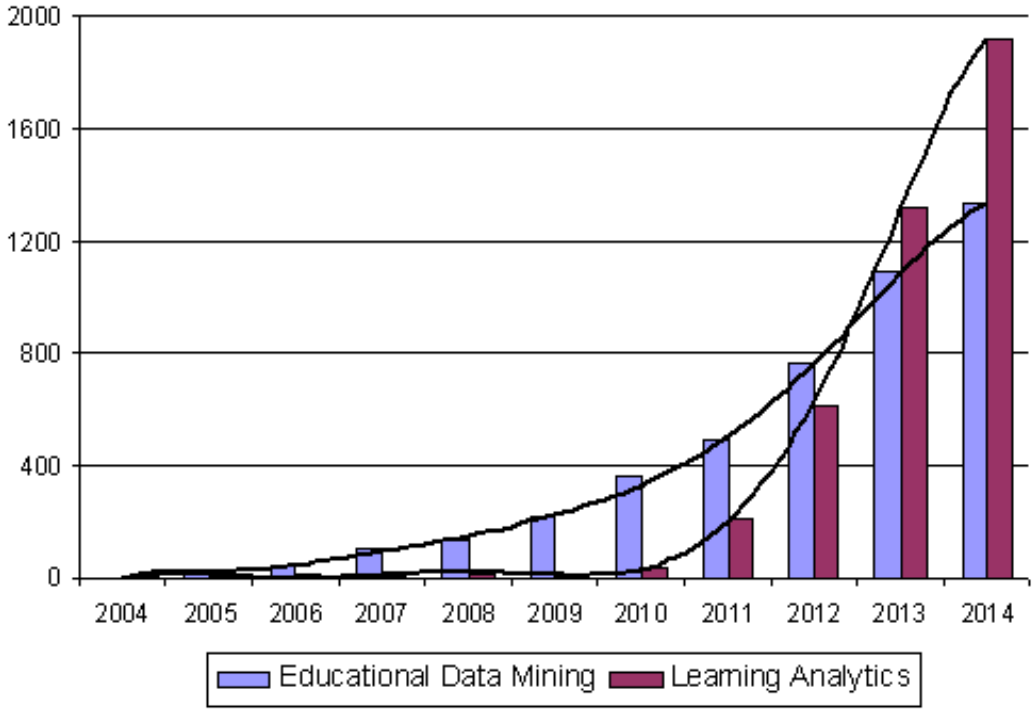


FIGURE C.3: Evolution of EDM and LA references in Google Scholar (May, 2015).

According to Siemens and Baker (2012), it is possible to identify five key distinctions between EDM and LA. These are:

- Discovery: in EDM, researchers are interested in automated discovery, and leveraging human judgment is a tool for that; in LA it is quite the opposite, leveraging human judgement is the aim.
- Reduction and holism: EDM reduces systems to components and explores them and their relationships, while LA wants to understand whole systems.

- Origins: EDM is rooted in educational software and student modelling; in contrast, LA origins are related to the semantic web, “intelligent curriculum", outcome prediction and systemic interventions.
- Adaption and personalization: EDM performs automated adaptation, whereas LA informs and empowers instructors and students.
- Techniques and methods: EDM employs more techniques and methods of classification, clustering, Bayesian modelling, relationship mining, discovery with models, and visualization; while LA focuses on social network analysis, sentiment analysis, influence analysis, discourse analysis, learner success prediction, concept analysis and sense-making models.

According to the above authors, these differences represent broad trends in each community and, as a consequence, they do not define the corresponding scopes. A similar idea is expressed in Baker and Inventado (2014), where it is stated that “the overlap and differences between the communities is largely organic, developing from the interests and values of specific researchers rather than reflecting a deeper philosophical split".

Bienkowski et al. (2012) consider that LA covers more disciplines than EDM does. In addition to computer science, statistics, psychology and the learning sciences, LA is related to information science and sociology. Therefore, even if the border between both fields is fuzzy and their differences are partly based on their origins and trends, they are still significant for these authors. Moreover, as upheld in Siemens and Baker (2012), the co-existence of both research communities leads to a more diverse and relevant contribution to society. Consequently, communication and competition between both should be encouraged.

6. Challenges and new trends

In spite of the high expectations and the relatively extensive literature on EDM and LA, they are relatively new fields of research and, as a result, several issues still need to be addressed. In addition, technological progress is driving us to the era of Big Data, which represents an important paradigm shift and offers multiple opportunities.

An important barrier to the implementation of EDM and LA methodologies is the lack of knowledge (Wolf et al., 2014), both theoretical and practical, among a significant proportion of instructors and managers with regard to employing the required tools, correctly understand the outputs, drawing the appropriate conclusions or deciding which actions to take. In order to mitigate this problem, it is important to increase acceptance and develop a data-driven culture in educational environments (Romero and Ventura, 2013). Researchers are already helping in this transition by disseminating their results, collaborating with a high number of instructors and/or students to assess their proposals (e.g., García et al. (2011)) and detailing their experiments (data, methods, etc.). As shown in this article, there are numerous tools to facilitate data analysis, but many have been implemented in small experiments. We will only be able to obtain more satisfactory and generic results by analyzing more students, courses and institutions.

Another significant barrier, discussed in Greller and Drachsler (2012), is related to ethics and personal privacy. Ethics must be taken into account in all stages, from data gathering to the interpretation of outputs and decision making, for instance, by avoiding statements that could lead to discriminatory treatments when working with gender, social status, race, home country, religious beliefs, ideology or disability. Similarly, issues related to the ownership of student data, which differ from country to country, need to be considered.

Numerous applications of EDM and LA methodologies in online environments deal with the use of Big Data in educational environments. Big Data refers to data with sizes beyond the ability of common software tools to capture, store, manage and process in a reasonable amount of time (Snijders et al., 2012). The main differences between Big Data and Analytics are volume, speed and variety (McAfee et al., 2012). In the past, obtaining, storing and processing data was an expensive and time-consuming procedure. Consequently, most studies attempted to draw conclusions from a sample of individuals that could be generalized to a population. However, current technology enables researchers to work with much more individuals and variables, obtaining richer information and insights. It leads to faster and more robust results, which should translate into more efficient decisions. The combination of Big Data and LA constitutes a promising field for governments (e.g., Johnson et al. (2013)) and universities (http://openthoughts-analytics.blogs.uoc.edu/) to explore.

Also, Massive Open Online Courses (MOOCs), typically managed by recognized instructors from prestigious universities, represent a new and prominent research topic. Besides being a marketing strategy for universities, they enable students from around the world to take modern and diverse courses for free, which helps to reduce the educational-opportunity divide associated with economic inequalities. According to Siemens (2013), the term “MOOC" is employed to refer to two different concepts: connectivist MOOCs (cMOOCs), which are based on a connectivist pedagogical model that uses freely available online resources, and edX MOOCs (xMOOCs), which replicate online the traditional model in which instructors share their knowledge and experience, and grade student assignments. The popularity of xMOOCs has been growing since 2012, when several large universities started to offer them. Coursera, edX, or Class2Go are some well-known platforms. These courses are usually characterized by a large number of enrolled students, which generates a scalability challenge (Kay et al., 2013), very high dropout rates and very different patterns of participation (Clow, 2013). Nevertheless, several authors agree that even if a high dropout rate raises concerns about a course, it is needed to take into account two elements: (a) the first exploratory phase, where students assess the content, structure and resources, and may decide not to continue; and (b) the diverse objectives, learning styles or schedules of students. Therefore, non-completion cannot be directly interpreted as a failure or problem. The maximum potential of EDM and LA in MOOCs stems from two facts: the diversity of students and the extremely high student-instructor rate. Participants may have different origins, backgrounds, maturity, experience, education levels, language skills, objectives, needs and learning styles, among others. This, in turn, suggests the relevance of personalizing courses. However, given the student-instructor rates, this is impossible without automated systems. Despite the fact that research on this topic is just emerging and that current MOOC platforms provide limited data storage, a few interesting works on adaptive MOOCs (aMOOCs) already exist. For example, Daradoumis et al. (2013) propose the use of software agents to improve and personalize management, delivery and evaluation. Agents could help to redesign MOOCs for future cohorts by gathering information on usage patterns, navigation, problematic content areas, tool usage, student profiling, etc. Regarding content, learning/prediction algorithms could be applied by agents to dynamically adjust course content to suit each participant’s profile. Furthermore, agents could be also employed to improve automated testing by adjusting assignment questions according to the participant’s educational level. Sonwalkar (2013) describes the development of the first aMOOC platform, which is implemented using Amazon Web Services’ cloud architecture. A case study of a course of molecular dynamics is analyzed. It considers different learning strategies based on five pedagogies (apprentice, incidental, inductive, deductive and discovery). The adapted learning path of each student is set at the beginning with a diagnostics quiz. As Clark (2013) critically notes, many MOOCs may be described as a set of linear sequential videos, quizzes and assessments reviewed automatically or by peers, while big companies like Google or Amazon employ algorithmic approaches to tailor searches, ads and recommend purchases. Therefore, aMOOCs are expected to become the focus of much more research attention over the coming years.

7. Conclusions

Educational Data Mining (EDM) and Learning Analytics (LA) are both relatively new and promising fields of research that aim to improve educational experiences by helping stakeholders (instructors, students, administrators and researchers) to make better decisions using data. Their growth has been boosted by increasing computer capacity to store and analyze huge amounts of data and the availability of statistical, machine-learning and data-mining methods and techniques.

Online environments are a highly important area of application. On the one hand, they continuously generate data from a number of events such as reading files or participating in forums, with different formats and levels of hierarchy. At the same time, online courses have higher dropout rates than traditional courses. EDM and LA are mainly employed to monitor students and groups (allowing the identification of students that are likely to dropout or fail, or that are not contributing enough in collaborative activities), suggest changes in course structure and tailor learning experiences (recommending material according to motivations and skills, for instance). There is a wide variety of methods and techniques adapted from other disciplines or specially designed to analyze educational data. Numerous similarities exist between both fields of research, such as goals, methodologies and techniques. However, there are several differences, attributable mostly

to their origins and trends. The co-existence of their respective scientific communities leads to competition with positive effects on society.

Despite the high expectations and the amount of works on EDM and LA, their application in educational environments still comes up against some important barriers, such as the lack of a data-driven culture and of fast, comprehensive and easy-to-use and understand tools that could be integrated in the most popular LMSs.

In the era of Big Data, the combination of the current capacity to capture, store, manage and process data in a reasonable amount of time, and data from online learning environments represents an opportunity for researchers into EDM and LA to better explore student learning processes and efficient ways to improve them. An important application is in MOOCs, where data from thousands of students can be employed to redesign courses for future students, relying on navigation and tool usage for example. A much more challenging approach consists in the development of adaptive MOOCs, in which the courses are automatically personalized according to student profiles (needs, objectives, background, country, learning style, etc.) and performance. This is a relatively new research topic that is currently getting much attention from both researchers and companies.

References

Antonenko, P. D., S. Toy, and D. S. Niederhauser (2012). "Using cluster analysis for data mining in educational technology research". In: *Educational Technology Research and Development* 60.3, pp. 383–398.

Baker, R. S. and P. S. Inventado (2014). "Educational data mining and learning analytics". In: *Learning analytics*. Springer, pp. 61–75.

Baker, R. S. J. D., A. T. Corbett, and A. Z. Wagner (2006). "Human classification of low-fidelity replays of student actions". In: *Proceedings of the Educational Data Mining Workshop at the 8th International Conference on Intelligent Tutoring Systems*. Vol. 2002, pp. 29–36.

Baker, R. S. J. D., T. Barnes, and J. E. Beck (2008). *Proceedings of the 1st International Conference on Educational Data Mining*. Montreal, Quebec, Canada.

Barnes, T., M. Desmarais, C. Romero, and S. Ventura (2009). *Proceedings of the 2nd International Conference on Educational Data Mining*. Cordoba, Spain.

Bienkowski, M., M. Feng, and B. Means (2012). "Enhancing teaching and learning through educational data mining and learning analytics: An issue brief". In: *US Department of Education, Office of Educational Technology*, pp. 1–57.

Clark, D. (2013). "Adaptive MOOCs". In: *Retrieved from* http://www.cogbooks.com/white-papers-AdaptiveMOOCs.html.

Clow, D. (2013). "MOOCs and the funnel of participation". In: *Proceedings of the Third International Conference on Learning Analytics and Knowledge*. ACM, pp. 185–189.

Daradoumis, T., I. Rodríguez-Ardura, J. Faulin, and F. J. Martínez-López (2010a). "CRM Applied to Higher Education: Developing an e-Monitoring System to Improve Relationships in e-Learning Environments". In: *International Journal of Services Technology and Management* 14.1, pp. 103–125.

Daradoumis, T., A. A. Juan, F. Lera-López, and J. Faulin (2010b). "Using collaboration strategies to support the monitoring of online collaborative learning activity". In: *Technology Enhanced Learning. Quality of Teaching and Educational Reform*. Springer, pp. 271–277.

Daradoumis, T., R. Bassi, F. Xhafa, and S. Caballé (2013). "A review on massive e-learning (MOOC) design, delivery and assessment". In: *P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), 2013 Eighth International Conference on*. IEEE, pp. 208–213.

Desmarais, M. C. (2012). "Mapping question items to skills with non-negative matrix factorization". In: *ACM SIGKDD Explorations Newsletter* 13.2, pp. 30–36.

García, E., C. Romero, S. Ventura, and C. De Castro (2011). "A collaborative educational association rule mining tool". In: *The Internet and Higher Education* 14.2, pp. 77–88.

Greller, W. and H. Drachsler (2012). "Translating learning into numbers: a generic framework for learning analytics". In: *Educational technology & society* 15.3, pp. 42–57.

Jeong, H. and G. Biswas (2008). "Mining student behavior models in learning-by-teaching environments". In: *Proceedings of the 1st International Conference on Educational Data Mining*.

Juan, A. A., T. Daradoumis, J. Faulin, and F. Xhafa (2009a). "A data analysis model based on control charts to monitor online learning processes". In: *International Journal of Business Intelligence and Data Mining* 4.2, pp. 159–174.

Juan, A. A., T. Daradoumis, J. Faulin, and F. Xhafa (2009b). "SAMOS: a model for monitoring students' and groups' activities in collaborative e-learning". In: *International Journal of Learning Technology* 4.1-2, pp. 53–72.

Kay, J., P. Reimann, E. Diebold, and B. Kummerfeld (2013). "MOOCs: So many learners, so much potential". In: *Technology* 52.1, pp. 49–67.

Kinnebrew, J. and G. Biswas (2012). "Identifying learning behaviors by contextualizing differential sequence mining with action features and performance evolution". In: *Educational Data Mining 2012*.

Larusson, J. A. and B. (Eds.) White (2014). *Learning Analytics: from Research to Practice*. Springer.

Lee, J. I. and E. Brunskill (2012). "The impact on individualizing student models on necessary practice opportunities". In: *Proceedings of the 5th International Conference on Educational Data Mining*.

Lera-López, F., J. Faulin, A. A. Juan, and V. Cavaller (2009). "Monitoring students' activity and performance in online higher education: A European perspective". In: *Monitoring and Assessment in Online Collaborative Environments: Emergent Computational Technologies for E-Learning Support*, p. 131.

Long, P., G. Siemens, G. Conole, and D. Gašević (2011). *Proceedings of the 1st International Conference on Learning Analytics and Knowledge*. Banff, Alberta, Canada.

Marquès, J. M., D. Lazaro, A. A. Juan, X. Vilajosana, M. Domingo, and J. Jorba (2013). "PlanetLab@ UOC: A real lab over the Internet to experiment with distributed systems". In: *Computer Applications in Engineering Education* 21.2, pp. 265–275.

McAfee, A., E. Brynjolfsson, T. H. Davenport, D. J. Patil, and D. Barton (2012). "Big data". In: *The management revolution. Harvard Bus Rev* 90.10, pp. 61–67.

Palazuelos, C., D. García-Saiz, and M. Zorrilla (2013). "Social network analysis and data mining: an application to the e-learning context". In: *International Conference on Computational Collective Intelligence*. Springer, pp. 651–660.

Peña-Ayala, A. (2013). *Educational data mining: applications and trends*. Vol. 524. Springer.

Romero, C. and S. Ventura (2006). *Data mining in e-learning*. Vol. 4. Wit Press.

— (2007). "Educational data mining: a survey from 1995 to 2005". In: *Expert systems with applications* 33.1, pp. 135–146.

— (2010). "Educational data mining: a review of the state of the art". In: *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 40.6, pp. 601–618.

— (2013). "Data mining in education". In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 3.1, pp. 12–27.

Romero, C., S. Ventura, M. Pechenizkiy, and R. S. J. D. Baker (2010). *Handbook of educational data mining*. CRC Press.

Siemens, G. (2013). "Massive open online courses: Innovation in education". In: *Open educational resources: Innovation, research and practice* 5.

Siemens, G. and R. S. J. D. Baker (2012). "Learning analytics and educational data mining: towards communication and collaboration". In: *Proceedings of the 2nd international conference on learning analytics and knowledge*. ACM, pp. 252–254.

Snijders, C., U. Matzat, and U. Reips (2012). "Big Data: big gaps of knowledge in the field of internet science". In: *International Journal of Internet Science* 7.1, pp. 1–5.

Sonwalkar, N. (2013). "The first adaptive MOOC: a case study on pedagogy framework and scalable cloud architecture—Part I". In: *MOOCs Forum*. Vol. 1. P, pp. 22–29.

Tane, J., C. Schmitz, and G. Stumme (2004). "Semantic resource management for the web: an e-learning application". In: *Proceedings of the 13th international World Wide Web conference on Alternate track papers & posters*. ACM, pp. 1–10.

Trčka, N., M. Pechenizkiy, and W. V. D. Aalst (2010). *Process mining from educational data*. Chapman & Hall/CRC.

Ueno, M. (2004). "Online outlier detection system for learning time data in e-learning and its evaluation". In: *CATE*, pp. 248–253.

Wolf, M. A., R. Jones, S. Hall, and B. Wise (2014). "Capacity enablers and barriers for learning analytics: implications for policy and practice". In: *Alliance for Excellent Education*.

Yadav, S. K. and S. Pal (2012). "Data mining: A prediction for performance improvement of engineering students using classification". In: *World of Computer Science and Information Technology Journal* 2.2, pp. 51–56.

C.2 A simheuristic for the heterogeneous site-dependent asymmetric VRP with stochastic demands

Laura Calvet[1], Adela Pagès-Bernaus[1], Oriol Travesset-Baro[2] and Angel A. Juan[1]

1. Computer Science Dept., Open University of Catalonia - IN3, Castelldefels, Spain
e-mail: {lcalvetl, apagesb, ajuanp}@uoc.edu
2. Observatori de la Sostenibilitat d'Andorra (OBSA), Sant Julià de Lòria, Andorra
e-mail: otravesset@obsa.ad

Abstract

Rich Vehicle Routing Problems (RVRPs) refer to complex and realistic extensions of the classical Vehicle Routing Problem. They constitute a hot topic in logistics due to their high number of relevant applications. This work focuses on a RVRP with the following characteristics: *(a)* heterogeneous fleet of vehicles, *(b)* site-dependency, i.e., not all types of vehicle can reach all customers, *(c)* asymmetric costs, and *(d)* stochastic demands. We formally define the problem and describe real-life applications. Our main contribution is a simheuristic-based methodology including a Successive Approximations Method for solving it. A computational experiment is carried out to illustrate the proposed methodology. Moreover, the suitability of considering a simheuristic approach is analyzed.

Keywords: Simheuristics, Successive Approximations Method, Heterogeneous VRP, Site-dependent VRP, Metaheuristics, Stochastic Optimization Problems.

1. Introduction

In our globalized and dynamic economy, transport constitutes the backbone of complex and large supply chains that require the fast, economical and reliable flow of goods. In most countries, road transport is the most used system. In addition to contribute to Gross Domestic Product and employment, the correct functioning of this sector is essential for other sectors, having an important effect on company competitiveness. While the activity of this sector is growing due to the increase of good demands (as a consequence of the increase of population and purchasing power), infrastructures are limited. Moreover, road transport causes congestion, accidents, noise, pollution, environmental impacts, and dependency on imported fossil fuels. Therefore, the development of efficient optimization methods is required.

The delivery of goods has always been a challenging problem for the Operations Research community. The popular Vehicle Routing Problem (VRP) (where routes are built to visit a set of customers with uncapacitated vehicles, minimizing distance-based costs) is the most basic example. This problem has been extended in many directions to introduce realistic characteristics. A classification and review on these rich problems can be found in Caceres et al. (2014).

We study the Heterogeneous Site-dependent Asymmetric VRP with Stochastic Demands (HSAVRP-SD). Its main goal is to minimize the costs associated to the distribution of goods in such a way that all demands are satisfied. It considers a number of vehicles that may have different loading capacities (i.e., a heterogeneous fleet). This diversity usually comes from two facts: different customers and locations may require different types of vehicle (e.g., due to narrow roads, available parking spaces, and vehicle weight restrictions), and the vehicle acquisitions may be made in different times and places. Related to this characteristic, the HSAVRP-SD describes a scenario where some customers cannot be accessed with all types of vehicle, which is known as site-dependency. Regarding the cost matrix, we relax the classical assumption about its symmetry, since there can be cost differences associated to the direction of a route (for instance, differences between driving uphill or downhill in mountainous regions). Finally, we account for uncertainty in demands by modeling them as random variables following specific probability distributions (either empirical or theoretical ones).

The HSAVRP-SD has several real-life applications. A typical example is the fuel oil distribution which can be associated to petrol station replenishment or to the delivery of domestic heating oil. Generally, in these cases, the exact demand required by a customer is not known until the time of the delivery, and cost between nodes (based on energy consumption) is asymmetric due to the presence of important road grades. A review of the petrol station replenishment problem

is presented in Cornillier et al. (2012). The optimization of domestic heating oil distribution has been less studied, even though the high dependence on heating oil of some isolated regions. Once again, it could be of particular interest in mountainous regions where in absence of a gas pipeline, domestic oil is frequently the predominant fuel for heating.

Being the analyzed problem a NP-hard and Stochastic Combinatorial Optimization Problem (SCOP) (Bianchi et al., 2009), we propose a simheuristic-based methodology for addressing it. Simheuristics (Juan et al., 2015a) is a relatively new approach for solving SCOPs in a natural way by combining metaheuristics and simulation techniques. While the former search for promising solutions, the latter assess their performance in a stochastic environment. We apply an Iterated Local Search (ILS) metaheuristic (Lourenço et al., 2010), which employs the Successive Approximations Method (SAM) (Juan et al., 2014c) for creating solutions, and Monte Carlo Simulation (MCS) techniques. The main contributions of this paper are: *(1)* a simple yet powerful methodology for solving the HSAVRP-SD; and *(2)* a computational experiment carried out to illustrate the methodology and assess the need of a simheuristic approach.

2. Literature Review

Heterogeneous Site-dependent Asymmetric VRPs

Nag et al. (1988) is the first work to analyze site-dependencies, which propose simple heuristics. Other authors (Chao and Liou, 2005; Cordeau and Laporte, 2001; Pisinger and Ropke, 2007) also address this characteristic by applying metaheuristics. Regarding asymmetric costs, Herrero et al. (2014) present a hybrid algorithm including biased randomization techniques and several local searches for solving the Asymmetric and Heterogeneous VRP. In Yusuf (2014) the authors suggest a three-phase heuristic for the Multi-depot Heterogeneous, Site-dependent and Asymmetric VRP. The characteristic of heterogeneity has been much more studied than the others; Hoff et al. (2010) provide a comprehensive review.

Stochastic VRPs (SVRPs)

Despite the fact that SVRPs have not been so extensively studied as the deterministic counterparts, there are some interesting works related to this article. For instance, Tillman (1969) is considered the first work. It expands the CWS heuristic (Clarke and Wright, 1964) to address the Multi-Depot Vehicle Routing Problem with Stochastic Demands, which follow Poisson distributions. The concept of route failure (i.e., when the demand of the customer being visited by a given vehicle exceeds its remaining capacity) is proposed in Dror and Trudeau (1986).

A review of the main works on SVRPs (Gendreau et al., 1996) classifies them according to the stochasticity source (customers, demands and/or times). Yang et al. (2000) propose anticipating potential route failures by incorporating preventive breaks in the design of routes. Bianchi et al. (2006) compare the performance of a few metaheuristics (Simulated Annealing, Tabu Search, ILS, Ant Colony Optimization and Evolutionary Algorithm) for solving the VRP with Stochastic Demands (VRP-SD). It is also worthwhile mentioning the work of Bianchi et al. (2009), which provides a comprehensive survey on metaheuristics for SCOPs.

Many works have tackled SVRPs using simheuristics. The VRP-SD is addressed in Juan et al. (2011b), which introduces the use of safety stocks. It consists of reducing the vehicle capacities, multiplying them by a number between 0 and 1, only for designing the routes. Thus, the remaining capacity is kept as a reserve in case real demands are higher than expected. An improved version is presented in Juan et al. (2013b), where the benefits of parallel and distributed computing are studied.

3. Definition of the Problem

The HSAVRP-SD is defined over a complete graph $G = (N, A)$, where $N = \{0, 1, \ldots, n\}$ is a set of nodes representing the depot (node 0) and the n customers (nodes 1 to n). Each node $i \in N$ has associated a demand D_i, which is a random variable following a given probability distribution. The actual demand of a specific customer is only known when a vehicle visits her/him. It is assumed that the depot has no demand. The set $A = \{(i, j) : i, j \in N, i \neq j\}$ contains the arcs connecting each pair of nodes. Moreover, there is a set $F = \{1, ..., m\}$ referring to the types of vehicle. For each type $o \in F$, there are p_o available vehicles, the parameter Q_o represents the maximum load

that a vehicle can carry, and U_o ($U_o \subseteq N \setminus 0$) denotes the set of customers that can be served. Each arc has associated a cost c_{ij}^o that depends on the the type of vehicle. The cost of a route is the sum of the costs of its arcs and a fixed cost for using a vehicle (f_o).

The goal is to design routes that satisfy all demands and minimize the total costs while satisfying the constraints previously introduced and the following ones: *(a)* each vehicle starts and ends its route at the depot; and *(b)* each customer is visited by just one vehicle.

4. Our Methodology

The methodology we propose is a simheuristic procedure combining the ILS metaheuristic and MCS techniques. Simheuristics efficiently solve SCOPs in a natural way by extending competitive metaheuristics for the deterministic problem (usually much more studied) with simulation techniques. The ILS metaheuristic is highly popular for addressing a wide range of problems in routing, scheduling, finance, etc. It is relatively easy to understand and to implement because of its modularity. MCS enables the assessment of solutions in a stochastic environment by following these steps: *(1)* simulate a set of scenarios (where each scenario is created by generating a value for each random variable); and *(2)* compute the mean value of a performance measure. For building solutions, the SAM procedure is used. The description of our methodology is explained below and summarized in Figure C.1 and Algorithm 1.

Algorithm 1 The SAM procedure

```
1:  procedure BUILDSOLUTION(customers, vehicles)
2:      globalSol ← empty
3:      nonServedCust ← customers
4:      while nonServedCust ≠ empty do
5:          vehType ← selectType(vehicles)
6:          compatCust ← getCompatibleCust(nonServedCust, vehType)
7:          sol ← solveHoSAVRP(compatCust, vehType)
8:          routes ← getRoutes(sol)
9:          numVehOfTypeK ← numberOfVehicle(vehType)
10:         if numberOfRoutes > numVehOfTypeK
11:             routes ← SelectRoutes(numVehOfTypeK, Random)
12:         end if
13:         globalSol ← addRouteToSol(routes, globalSol)
14:         vehicles ← deleteUsedVehicles(vehicles)
15:         nonServedCust ← extractCustomers(nonServedCust, globalSol)
16:     end while
17:     return globalSol
18: end procedure
```

The inputs are the HSAVRP-SD instance, where each demand is modeled as a random variable following a specific probability distribution, and a set K of values used to determine safety stocks. Their use leads to lower costs due to route failures (which are the costs of going from the customer being served to the depot to refill and come back to complete the delivery). However, it may also increase the number of routes needed, increasing the deterministic costs (those obtained considering that demand variances are 0). Consequently, it is required to test different values and compare expected total costs.

The algorithm starts by selecting the first value $k \in K$ and transforming the original instance into a deterministic one replacing stochastic demands by their means. Additionally, the capacities are reset to: $Q_o = (1-k)Q_o \ \ (\forall o \in F)$. The next step consists in building an initial solution (*initSol*) for the new instance and estimating the associated total costs using MCS techniques with a short number of scenarios. Afterwards, a base solution (*baseSol*) is constructed by cloning *initSol*, and a list of solutions (*bestSols*) is created, which will store the best stochastic solutions (i.e., those with the lowest expected total cost). Initially, the list includes (*initSol*). Then, a new solution (*newSol*) is obtained by perturbing *baseSol*, which involves removing a random number of routes and repairing it. If the former has lower total costs (i.e., costs in the deterministic environment), it replaces *baseSol*, the total costs in the stochastic environment are estimated with a short MCS, and *bestSols* is updated. On the other hand, if (*newSol*) is not better than (*baseSol*), an acceptance criterion is

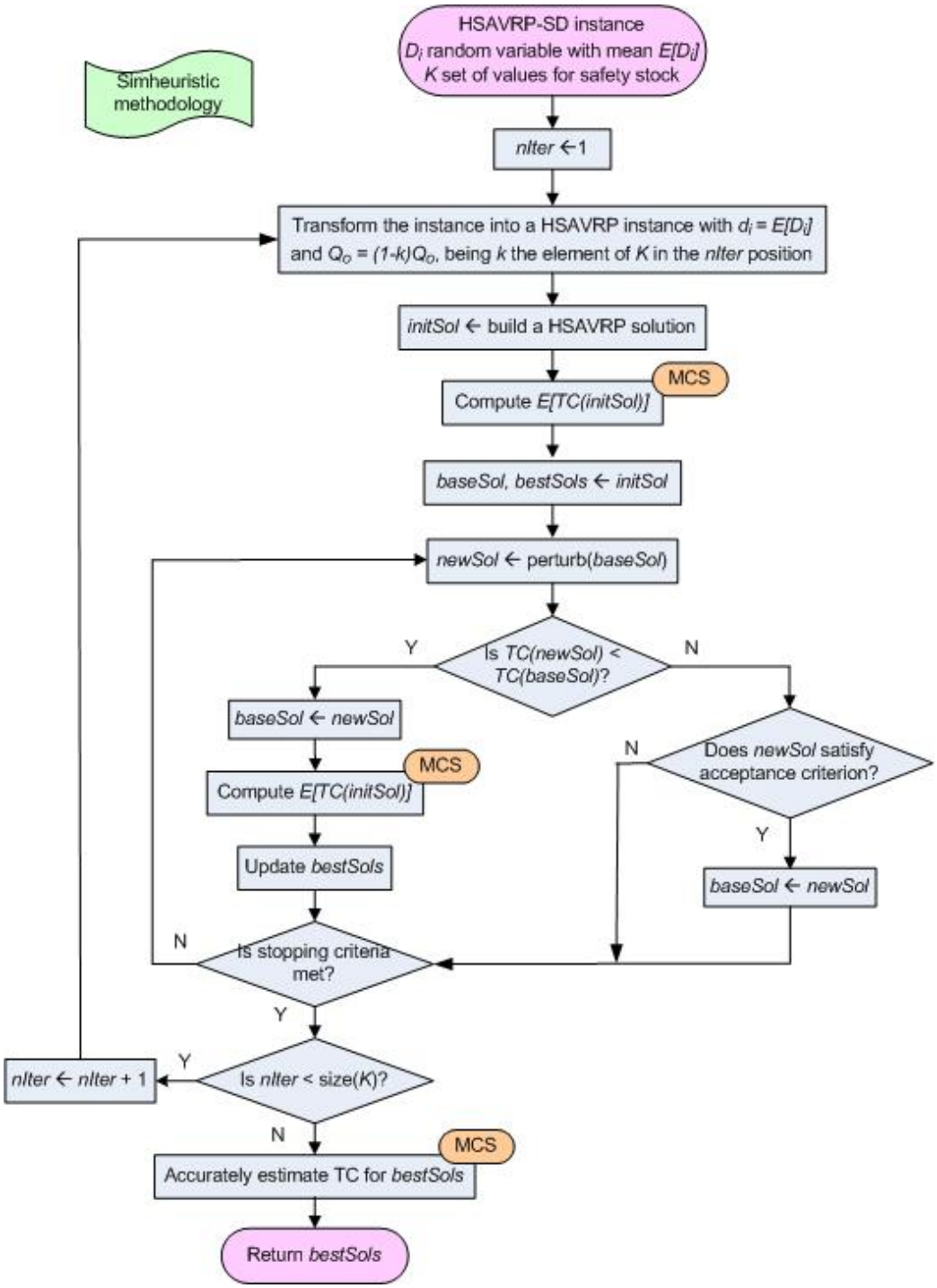


FIGURE C.1: Flowchart of our methodology.

checked to decide whether the base solution is replaced. We use a Demon-like acceptance criterion (Talbi, 2009), which allows the base solution to be deteriorated if no consecutive deteriorations take place and the degradation does not exceed the value of the last improvement. By doing this, the algorithm avoids getting stuck in a local optima. This procedure is repeated to visit different solutions until a stopping criteria is met. At this point, the algorithm is re-initialized with another value of K. When all values have been tested, the total costs of *bestSols* are accurately estimated using MCS with a larger number of scenarios. Finally, the list is returned.

Regarding the building of solutions, the SAM procedure is implemented. It can be described as follows. The procedure receives one list of customers and one of available vehicles. First, an empty global solution is created, and the list of customers is copied into a list of non-served

customers. While this list is not empty, the next steps are taken. A vehicle type not used yet is selected and those customers not compatible with the selected vehicle are removed from the list. Then the problem is transformed into an Homogeneous SAVRP (HoSAVRP) with no limitation on the number of vehicles that is solved with a state-of-the-art algorithm.

If the solution provided reports more routes than the number of available vehicles of the current type, some routes are discarded. This partial solution is included in the global solution. The last instructions inside the while loop update the list of available vehicles and the list of non-served customers. This process ends when all customers are assigned to a route. Finally, the global solution is returned.

The procedure for repairing solutions is exactly the same but receiving as inputs only those customers that remain to be included in a route and copying the perturbed solution into the global one when this is created.

Each HoSAVRP solution is constructed using the SR-GCWS-CS algorithm described in Juan et al. (2011a). It is based on the CWS heuristic and incorporates biased randomization techniques and cache and splitting techniques, which contribute to reduce computational times. We have adapted this algorithm in order to consider asymmetric costs. For this, the easy procedure of computing savings as the mean of the two savings associated to each pair of nodes (Gruler et al., 2015b) has been applied.

5. Computational Experiments

In order to test the simheuristic approach presented in the previous section, we have generalized a (randomly chosen) set of 4 classical CVRP instances. The original data can be downloaded from *Branch and Cut*. The same location of the nodes and demand is used. To include all the characteristics of the rich VRP, the instances have been modified in the following aspects.

Given that a cost perspective is taken as objective function, a fixed cost for using vehicle, f_o, and a variable cost, v_o, that multiplies the distance have been established. Therefore, the cost of arc $(i, j) \in A$, $c^o_{ij} = v_o d_{ij}$, where d_{ij} is the Euclidean distance. In order to account for asymmetric costs, the cost of an edge (i, j) is incremented by 10% if the y-coordinate of the destination node j is greater than the y-coordinate of the origin node j.

An heterogeneous fleet has been proposed, with three type of vehicles. Large vehicles have a capacity equal to the one used in the benchmark, and medium and small vehicles have a reduced capacity of 75% and 50% respectively. All vehicles can serve all customers except for customers belonging to a randomly selected sub-area in which we assume that large vehicles cannot access.

Without loss of generality we have chosen the demand of a particular customer, D_i, to follow a logNormal distribution, with expected value as the demand of the benchmark instance (d_i) and variance proportional to the expected value (κd_i). The results presented next are obtained with $\kappa = 0.1$.

All the assumptions were made considering realistic situations. The resulting instances can be provided to the interested reader.

Several measures will be computed for each solution. When a solution is evaluated with deterministic demands, the cost (Z^{det}) and the distance ($dist$) are shown. When a solution is assessed with stochastic demands, route failures may happen which increase the cost. Therefore, the expected cost (Z^{stoch}) and the percentage of expected route failures (r^{fail}) is displayed. Given that our strategy for searching alternative solutions with better performance in the stochastic world is to define safety stocks, this value is also included.

Test cases were run on a laptop with 4 cores at 2.6GHz. Experiments were run over 5 random seeds for 60 seconds except for instance A-n80-k10 which run for 300 seconds. The name of the instances indicate the number of nodes (after the letter n). Short MCS were run for 100 scenarios, and long simulations consisted of a sample of 10000.

Table C.1 compares the solution of the original CVRP instance with the current version HSAVRP with deterministic demands. When the SAM method is employed to solve the CVRP, Our Best Solution (OBS) shows to be competitive compared with the optimal (OPT) solution reported in the literature, with an average gap of 0.52%. With the solution of the HSAVRP we also report the composition of the fleet for each solution. We can observe that a mix fleet is used, motivated by the fact that some vehicles cannot access some customers. The performance of the deterministic solution is tested in the operational level with stochastic demands in Table C.2.

TABLE C.1: Test instances - Comparison between the original CVRP distance-based solution and the HSAVRP cost-based solution

	Original instance (CVRP)					Deterministic HSAVRP - OBS								
Instance	Veh. capac.	OPT	OBS	Gap	Veh. used	Avail. vehicles			Z^{det}	dist	Δ dist	Used vehicles		
					L	L	M	S				L	M	S
P-n40-k5	140	458	461.7	*0.81*	5	4	2	3	2318.3	559.5	*21.18*	3	2	1
B-n41-k6	100	829	833.7	*0.56*	6	5	6	6	3656.7	1075.2	*28.97*	4	4	0
B-n45-k5	100	751	754.0	*0.39*	5	4	3	3	2907.6	802.2	*6.40*	4	2	0
A-n80-k10	100	1763	1768.7	*0.32*	10	8	6	6	6809.2	2040.9	*15.39*	6	5	0

TABLE C.2: Comparison of solution under stochastic demand

	Deterministic HSAVRP				Stochastic demands HSAVRP					
Instance	Z^{det}	dist	Z^{sto} (1)	r^{fail}	Z^{det}	dist	Z^{sto} (2)	r^{fail}	Safety stock	Gap ((2)-(1))/(2)
P-n40-k5	2318.3	559.5	2320.7	3%	2318.3	560.5	2318.4	0%	100%	0.10%
B-n41-k6	3656.7	1075.2	4054.9	68%	3667.1	1078.6	3678.6	21%	99%	10.23%
B-n45-k5	2907.6	802.2	2972.9	56%	2912.4	804.1	2912.6	0%	99%	2.07%
A-n80-k10	6809.2	2040.9	7334.5	82%	6964.2	2063.0	7004.7	14%	98%	4.71%

A particular solution is tested under stochastic demands using MCS techniques, the same routine that we used in the simheuristic with a large sample. In Table C.2 we can observe how the expected cost of the deterministic solution increases on average a 4% and experiences a high percentage of route failures. This is mainly due to the fact that some routes has a filling rate of 99%, and small variations of the demand induce route failures. On the other hand, stochastic solutions show a filling rate more balanced among the routes, and the route failures decrease dramatically. Figure C.2 illustrates such situation for test case P-n40-k5. The expected cost of the stochastic solution outperforms that of the deterministic solution with an average of 4.28%, and results in lower variability of the costs (given the low percentage of route failures).

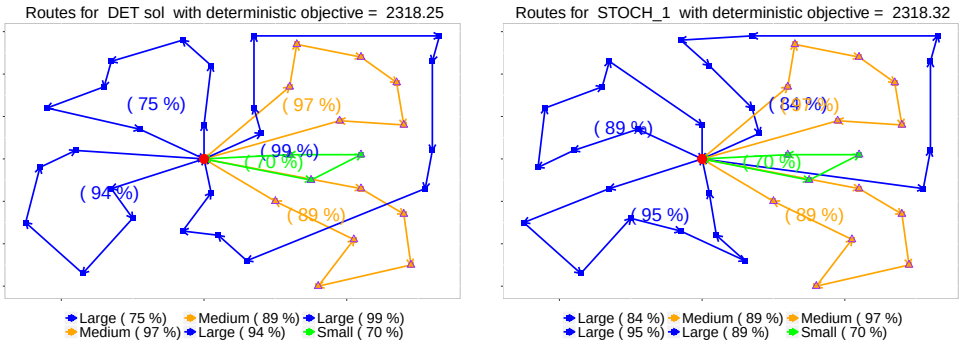


FIGURE C.2: Delivery routes for instance Pn40k5 - deterministic and stochastic solution

6. Conclusions

Road transport is increasingly relevant in our globalized economies, contributing to Gross Domestic Product and employment. On the other hand, it may lead to congestion, pollution, etc. In this work, we have focused on the Heterogeneous Site-dependent Asymmetric Vehicle Routing Problem with Stochastic Demands, which constitutes a complex and realistic problem. A solving methodology based on the simheuristic approach has been described. It employs a Successive Approximations Method. Results show that our methodology is able to solve the problem, and quantify the benefit of using simheuristics in stochastic environments.

Acknowledgments

This work has been partially supported by the Spanish Ministry of Economy and Competitiveness (TRA2013-48180-C3-P and TRA2015-71883-REDT), FEDER, the Catalan Government (2014-CTP-00001) and the Government of Andorra (ACTP022 - AND/2014).

References

Bianchi, L., M. Birattari, M. Chiarandini, M. Manfrin, M. Mastrolilli, L. Paquete, O. Rossi-Doria, and T. Schiavinotto (2006). "Hybrid metaheuristics for the vehicle routing problem with stochastic demands". In: *Journal of Mathematical Modelling and Algorithms* 5, pp. 91–110.

Bianchi, L., D. Marco, L. M. Gambardella, and W. J. Gutjahr (2009). "A survey on metaheuristics for stochastic combinatorial optimization". In: *Natural Computing* 8, pp. 239–287.

Branch and Cut. http://www.coin-or.org/SYMPHONY/branchandcut/VRP/data/. [Online; accessed March 2016].

Caceres, J., P. Arias, D. Guimarans, D. Riera, and A. A. Juan (2014). "Rich vehicle routing problem: a survey". In: *ACM Computing Surveys* 47.2. ISSN: 0360-0300.

Calvet, L., D. Wang, and A. A. Juan (submitted[a]). "A simheuristic algorithm for the stochastic multi-depot vehicle routing problem". In: *International Transactions in Operational Research*.

Chao, I. M. and T. S. Liou (2005). "A new tabu search heuristic for the site-dependent vehicle routing problem". In: *The Next Wave in Computing, Optimization, and Decision Technologies* 29, pp. 107–119.

Clarke, G. and J. Wright (1964). "Scheduling of vehicles from a central depot to a number of delivering points". In: *Operations Research* 12, pp. 568–581.

Cordeau, J. F. and G. Laporte (2001). "A tabu search algorithm for the site dependent vehicle routing problem with time windows". In: *Information Systems and Operational Research* 39.3, pp. 292–298.

Cornillier, F., F. Boctor, and J Renaud (2012). "Heuristics for the multi-depot petrol station replenishment problem with time windows". In: *European Journal of Operational Research* 220, pp. 361–369.

Dror, M. and P. Trudeau (1986). "Stochastic vehicle routing with modified savings algorithms". In: *European Journal of Operational Research* 23, pp. 228–235.

Gendreau, M., G. Laporte, and R. Séguin (1996). "Stochastic vehicle routing with modified savings algorithms". In: *European Journal of Operational Research* 88, pp. 3–12.

Gonzalez, S., D. Riera, A. A. Juan, M. G. Elizondo, and P. Fonseca (2012). "Proceedings of the 2012 Winter Simulation Conference". In: *SIM-RANDSHARP: a hybrid algorithm for solving the arc routing problem with stochastic demands*. Ed. by C. Laroque, J. Himmelspach, R. Pasupathy, O. Rose, and A. M. Uhrmacher, pp. 1–12.

Gruler, A., A. A. Juan, and M. Steglich (2015b). "Proceedings of the 2015 Metaheuristics International Conference. Agadir, Morocco". In: *A heuristic approach for smart waste collection management*.

Herrero, R., A. Rodriguez, J. Caceres-Cruz, and A. A. Juan (2014). "Solving vehicle routing problems with asymmetric costs and heterogeneous fleets". In: *International Journal of Advanced Operations Management* 6.1, pp. 58–80.

Hoff, A., H. Andersson, M. Christiansen, G. Hasle, and A. Løkketangen (2010). "Industrial aspects and literature survey: fleet composition and routing". In: *Computers and Operations Research* 37.9, pp. 2041–2061.

Juan, A. A., J. Faulin, R. Ruiz, B. Barrios, and S. Caballe (2010). "The SR-GCWS hybrid algorithm for solving the capacitated vehicle routing problem". In: *Applied Soft Computing* 10.1, pp. 215–224.

Juan, A. A., J. Faulin, J. Jorba, D. Riera, D. Masip, and B. Barrios (2011a). "On the use of Monte Carlo simulation, cache and splitting techniques to improve the Clarke and Wright savings heuristics". In: *Journal of the Operational Research Society* 62, pp. 1085–1097.

Juan, A. A., J. Faulin, S. Grasman, D. Riera, J. Marull, and C. Mendez (2011b). "Using safety stocks and simulation to solve the vehicle routing problem with stochastic demands". In: *Transportation Research Part C: Emerging Technologies* 19.5, pp. 751–765.

Juan, A. A., J. Faulin, J. Jorba, J. Caceres, and J. M. Marques (2013b). "Using parallel & distributed computing for real-time solving of vehicle routing problems with stochastic demands". In: *Annals of Operations Research* 207.1, pp. 43–65.

Juan, A. A., J. Faulin, J. Caceres, B. Barrios, and E. Martinez (2014c). "A successive approximations method for the heterogeneous vehicle routing problem: Analyzing different fleet configurations". In: *European Journal of Industrial Engineering* 8.6, pp. 762–788.

Juan, A. A., J. Faulin, S. Grasman, M. Rabe, and G. Figueira (2015a). "A review of simheuristics: extending metaheuristics to deal with stochastic optimization problems". In: *Operations Research Perspectives* 2, pp. 62–72.

Lourenço, H.R., O.C. Martin, and T. Stützle (2010). "Iterated local search: framework and applications". In: *Handbook of Metaheuristics*. Ed. by M. Gendreau and J.-Y. Potvin. Vol. 146. International Series in Operations Research & Management Science. Springer US, pp. 363–397.

Nag, B., B. Golden, and A. Assad (1988). "Vehicle routing: methods and studies". In: Amsterdam: Elsevier. Chap. Vehicle routing with site dependencies, pp. 149–159.

Pisinger, D. and S. Ropke (2007). "A general heuristic for vehicle routing problems". In: *Computers and Operations Research* 34.8, pp. 2403–2435.

Talbi, E.-G. (2009). *Metaheuristics: From design to implementation*. New Jersey: John Wiley & Sons.

Tillman, F. A. (1969). "The multiple terminal delivery problem with probabilistic demands". In: *Transportation Science* 3, pp. 192–204.

Yang, W. H., K. Mathur, and R. H. Ballou (2000). "Stochastic vehicle routing problem with restocking". In: *Transportation Science* 34, pp. 99–112.

Yusuf, I. (2014). “Solving multi-depot, heterogeneous, site dependent and asymmetric VRP using three steps heuristic”. In: *Journal of Algorithms and Optimization* 2.2, pp. 28–42.

Appendix D

Selected conference papers

D.1 SmartMonkey: A web browser tool for solving combinatorial optimization problems in real time

Xavier Ruiz[1], Laura Calvet[2], Jaume Ferrarons[2] and Angel A. Juan[2]

1. Incubio, Barcelona, Spain
e-mail: xavier.ruiz@incubio.com
2. Computer Science Dept., IN3-Open University of Catalonia, Barcelona, Spain
e-mail: {lcalvetl, ajuanp}@uoc.edu, jaume.ferrarons@gmail.com

Abstract

This paper introduces SmartMonkey, a novel web-browser approach for solving NP-hard combinatorial optimization problems in “real time" (usually a few seconds). Our approach makes use of randomized algorithms that are run in parallel on a set of independent machines available on the Internet. These machines do not need to be configured, and no client application needs to be installed on them. Instead, just by opening a web page in a web browser, the computational resources of the machine become available for the algorithms to be executed. Being a configuration-free approach, it offers a great advantage to end users, since they are relieved from the usually complex and time-consuming configuration tasks that characterize other distributed-computing approaches. Computational tests have been carried out using different algorithms for solving NP-hard combinatorial optimization problems in transportation and production scheduling. The results show that our approach allows obtaining near-optimal solutions in real time, which can be specially interesting for supporting decision-making processes, especially those in small and medium enterprises, in a wide range of application fields including logistics, transportation, smart cities, and manufacturing.

Keywords: Logistics & Transportation, Production, Combinatorial Optimization, Parallel and Distributed Computing, Randomized Algorithms, Metaheuristics.

1. Introduction

Internet plays an essential role in our society, permitting us not only to communicate and obtain information in ‘real-time’ but, as it will be discussed in this paper, also to solve complex decision-making problems in ‘real-time’. This is achieved by the use of Distributed and Parallel Computing Systems (DPCS), which allow the aggregation of multiple autonomous computing resources interacting to achieve a common goal (Coulouris et al., 2005). Under this general concept, there are a number of different paradigms. Grid Computing (Foster and Kesselman, 2003) appeared at the beginning of the 1990s decade, aiming to combine dispersed computational resources. It was initially used in scientific and research fields. Another relevant paradigm sharing the same goal is Cloud Computing (Armbrust et al., 2009). It emerged at the beginning of this century, focusing on a provider-client model in which users do not belong to any particular organization, only pay for the resources they use and can consume them at any time, without having to forecast their computing needs in a mid-large term. This paradigm eases Small and Medium Enterprises (SMEs) the acquisition of computer resources, since they do not longer need up-front investments or over-provisioning to face peak loads. More recent paradigms are Volunteer Computing (Sarmenta, 2001) and Desktop Grids (Cérin and Fedak, 2012). Both attempt to gather

surplus or idle computing resources. While the first focuses on end-users that voluntarily make available their resources for a third entity or project, the second utilizes resources from a given organization network. A successful project in the Volunteer Computing paradigm is SETI@home (http://setiathome.berkeley.edu) of BOINC ((Anderson, 2004)). However, it is usually a complex task to gather a considerable number of volunteers. Besides the savings, a particular interesting feature of Desktop Grids is that SMEs avoid sending private data to an external provider, and become more environmentally friendly.

The main contribution of this work is the description and testing of an efficient, flexible, and browser-based framework to facilitate access to computational resources (Berry, 2009) and, ultimately, solve Combinatorial Optimization Problems (COPs) in 'real time' (a few seconds). This framework allows the employment of new versions of web browsers (such as Google Chrome, Firefox, and Internet Explorer) as nodes in a cluster. The only required step is to visit a website. The embedded JavaScript code into this website enables the communication with the job dispatcher service. It may be considered a more scalable paradigm than traditional grid computing, since the connection of people is boosted by the fact that no third party software installation is required. Due to the relevance of COPs for SMEs and the amount of academic works proposing the implementation of DPCS for addressing them (Talbi, 2006; Talbi, 2009), we illustrate the working and the potential benefits of our approach by solving two classic NP-hard COPs in the fields of transportation (vehicle routing) and production (scheduling).

The rest of the paper is organized as follows: Section 2 provides an overview of Distributed and Parallel Computing Systems. Afterwards, Section 3 discusses the potential of DPCS-based approaches in solving real-life SMEs problems. The description of our browser-based platform is included in Section 4. The web-based approach developed to tackle COPs is explained in Section 5. Section 6 contains the numerical experiments carried out to analyse the efficiency of our approach, while Section 7 discusses the results of these experiments. Finally, Section 8 summarizes the main findings of this work.

2. Distributed and Parallel Computing Systems

Desktop computers have become affordable machines that most people use every day for both work and leisure. Despite their current capacity, numerous institutions and individuals require more computational resources to execute intensive problem-solving processes. In these cases, Distributed and Parallel Computing Systems constitute a useful approach. Multi-processors and/or multi-computers paradigms may be employed. A multi-processors schema refers to a set of physical processing units sharing a machine (mono-core CPU, multi-core CPU, or a combination of both). In this schema, a task represents a logical concept including instructions to be executed by an algorithm or application, while a process can be defined as a running instance of a computer program. A process might consist in different threads implementing one or more tasks. Tasks, threads or processes share a global memory system, on which their communications rely. On the other hand, a multi-computer schema presents a set of physical machines linked via net-work connections. These machines can be coupled geographically (in a supercomputing environment, for example) or in a more distributed environment (as in Cloud Computing). The main parallel paradigm is message passing, in which tasks and processes of different machines interchange data packets by sending and receiving messages to communicate.

Nowadays, there is no need for a user of designing and building a new computing infrastructure, since there are organizations which satisfy scalable computational demands at a reasonable price. Grids are mainly required in high-performance-computing scientific projects. Foster and Kesselman (2003) and Buyya and Venugopal (2005) provide an overview of this paradigm analysing several projects. In contrast, Cloud platforms constitute a solution for enterprises to obtain additional resources (public clouds), or to manage the resources owned (private cloud). Some examples of platforms providing these services are Amazon's Elastic Compute Cloud (http://aws.amazon.com/ec2) and Microsoft's Azure Services Platform (www.microsoft.com/windowsazure). Volunteer Computing models are mostly used for scientific and academic projects. Highly popular implementations are BOINC (Anderson, 2004), Condor (Litzkow et al., 1988) and Entropia (Chien et al., 2003). Examples of communities created to aggregate computational resources are Seti@HOME (http://setiathome.berkeley.edu) and Distributed.net (http://www.distributed.net). Finally, Desktop computing is employed by private organizations (Nadiminti and Buyya, 2005).

3. Relevance of DPCS for SMEs

SMEs are responsible for a significant part of the wealth generated in all developed economies. Often, they do neither possess advanced technical knowledge nor modern computational resources. However, a number of them could benefit from having more resources, for example to speed up intensive-computation processes or to obtain a higher performance. In order to access them, DPCS offer two alternatives: *(a)* to pay for using resources from an external provider (a cloud platform, for instance), which can be presented as virtual machines; and *(b)* to employ underutilized computer resources owned by the SME. This idea of aggregating idle or unused resources characterizes also Volunteer Computing Systems. The main difference between both paradigms is that while the latter is usually associated to dynamic (any user can freely enter and leave) and heterogeneous environments, an SME knows the characteristics and the availability of its machines. Obviously, their scalability is also more limited.

The alternative of using SME's underutilized resources presents several advantages. Firstly, SMEs do not have to send private information to servers of an external enterprise. Secondly, it is a cheaper solution since the SME does already have the resources. Finally, the energy consumption is reduced by seizing these resources, which could be still consuming otherwise (Cabrera, 2014). In this same direction, it can be argued that the environmental footprint of this alternative may be lower than that of a large digital warehouse, because the heat concentration is lower. These Desktop Grids Systems may be formed by personal computers with more computing capabilities than the required (standard computers in which employees mainly use word processors and spreadsheets, for instance) or that are not used during some specific days (weekends, holidays, etc.) or hours (night, midday, etc.). Moreover, resources from several SMEs may be gathered (Juan et al., 2013b) to build a larger computing system (Figure D.1). They can rely on a directory-of-resources service that keeps updated information of available computing resources. Once a user requires executing a process, he sends a query to this directory to select the resources and organize the tasks to perform. Once these tasks have been completed, the result is sent back to the user.

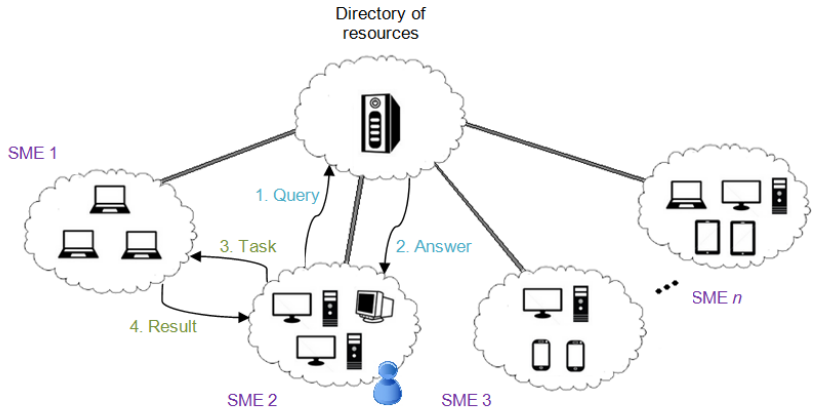


FIGURE D.1: Aggregating resources from one or several SMEs.

4. Description of the SmartMonkey Platform

SmartMonkey aims at facilitating the aggregation of a high number of computational resources – without any additional cost for the company– by seizing underutilized or idle resources. It is based on software already installed in most computers, web browsers. Using a modern version of some of the commonest (Google Chrome, Firefox, or Internet Explorer), it may integrate a computer into the computing network. The only action required is to visit a website with an embedded JavaScript code that enables the communication in real-time with a job dispatcher service. Each one of the jobs includes a piece of data and the computing task to perform. Additional steps such as downloading, installing, or setting up additional software are not required, which makes this option a very attractive one for most SMEs. Because the ease to add new resources, this approach can be considered highly scalable. As other Desktop Computing Systems, it has the advantage that does not require sending private data to an external enterprise or third party. Therefore, the described platform constitutes a flexible, simple, and scalable approach with multiple applications

in SMEs, which reduces the cost of acquiring additional computational capabilities. Figure D.2 shows the major difference between the BOINC framework and our web-based framework in terms of engagement. While our framework is just one step away to start processing, BOINC networks need a more complicated sequence of steps. Therefore, our configuration-free approach can turn out to be much more appealing to SMEs because its ease of use and fast scalability.

The platform architecture is the typical of a master-slave cluster. The system has been designed to free the master from computationally expensive tasks. For the experiments described later in this paper, a single master has been sufficient to handle all the workload. In a production environment, the system could easily scale to thousands of slaves or even further when considering other architectures like a multi-master environment, etc. In our case, the master was placed on a dedicated server located on a cloud provider (Softlayer). The slaves were located over 2 different locations: The UOC's Lab and the Incubio's offices. The execution process goes as follows. First, the end-user submits the task to be executed to the master. This task consists mainly in a set of Map and Reduce functions written in JavaScript, as well as their input dataset. The master is responsible of creating a list of jobs. Each job is composed of a chunk from the dataset and the source of code that has to be executed over each piece of data. The master delivers and ensures that jobs are evenly distributed. After each job is completed, the master receives the results and stores them in a file or a database depending on the execution flow given by the user. The master keeps track of the jobs that have been assigned and processed. Different measures handle unfinished jobs, errors or exceptions that could appear unexpectedly by either rescheduling the jobs or stopping the execution and reporting the error.

Some of the main benefits of this paradigm are cross-platform support and cluster scalability. Adding a slave to the system is as easy as adding a JavaScript snippet to any website and opening it with a modern browser. The loaded code is the responsible to communicate with the master and ask for new jobs to execute. As the code of each job is provided by the master, the code of the slave is in charge of receiving a new job, executing it, and then sending the results back to the master. This process is entirely written on JavaScript code, avoiding specificities of that language from specific browsers in such a way that it can run on a heterogeneous pool of different web browsers (such as Internet Explorer, Google Chrome and Firefox) as well as on different operating systems.

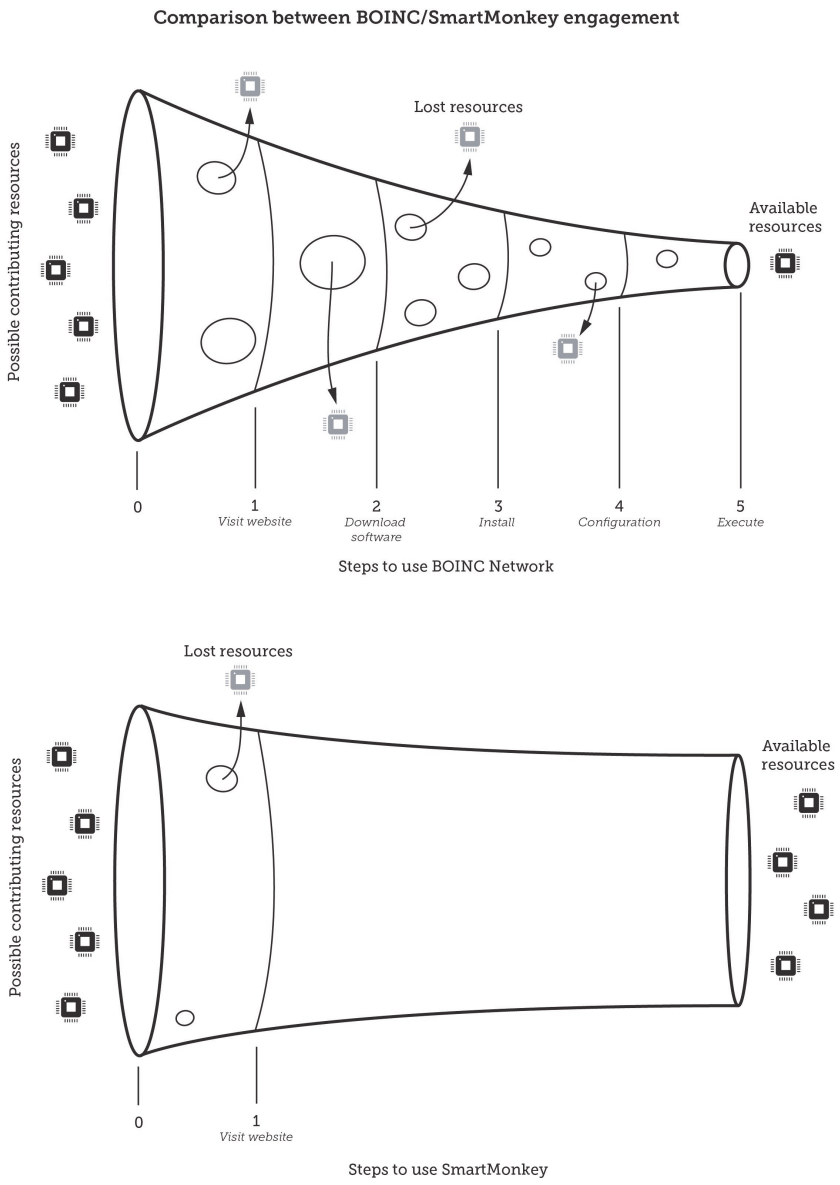


FIGURE D.2: Comparison between BOINC and SmartMonkey engagement process.

5. The Multi-agent Solving Approach

In sectors such as logistics and transportation, telecommunication, healthcare, finance, and production, a huge number of real-life decision-making processes can be modeled as COPs. Frequently, managers are required to take crucial decisions in real time. Designing routes to quickly provide medical assistance after a natural disaster, rescheduling flights because of some delays caused by unexpected circumstances, or reducing the risk of some savings in the stock market being affected by unpredicted events are a few examples. Despite the fact that exact methods exist for addressing COPs, they usually require a high amount of time to solve real-sized problem instances. As a consequence, approximate methods are widely implemented. Most of them are probabilistic, which means that their solution depends on the seed used for a pseudo-random number generator. It has been proved that the execution time that an algorithm needs to report high-quality solutions can be reduced depending on this seed (Juan et al., 2014d).

According to Talbi (2006) and Talbi (2009), DPCS are commonly employed to solve COPs. The typical approach in the related literature applies a master-slave scheme, in which a master or coordinator processor sends tasks to a set of slave processors in order to execute an intensive-computing process. Each slave is responsible for solving the same problem instance considering a different scenario, each one formed by a set of parameters and/or a seed. Once a slave has

completed its task, it sends the solution to the master that stores it. In the simplest version, there is no communication between slaves. Following this approach, we aim to execute multiple instances of the algorithm at the same time, each with a different seed. As shown in Figure D.3, each of these instances can be considered a cloned agent that is searching the solution space.

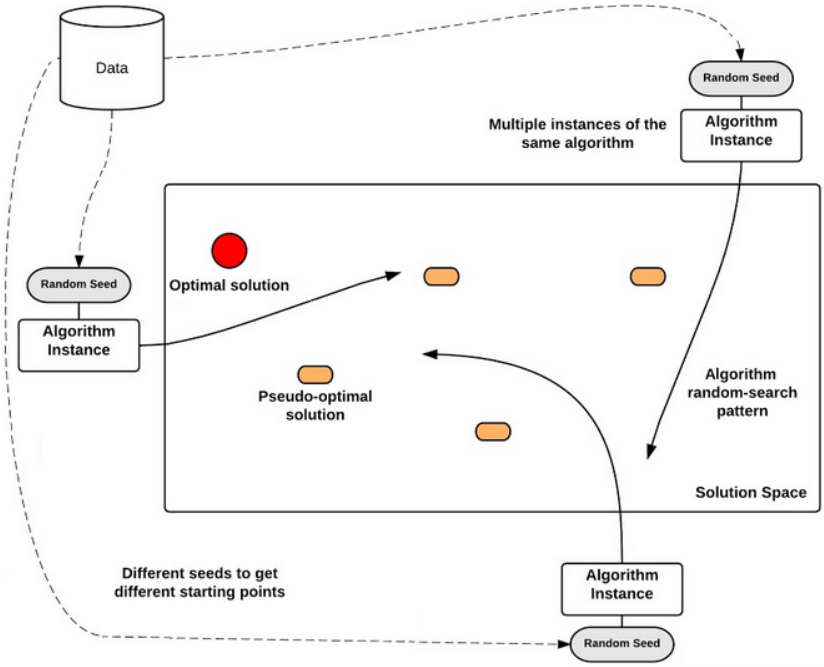


FIGURE D.3: A multi-agent approach for solving COPs.

6. Computational Experiments

In order to illustrate the benefits of the presented approach, two relevant NP-hard COPs have been addressed: the Capacitated Vehicle Routing Problem (CVRP) and the Permutation Flow-shop Sequencing Problem (PFSP). For each problem, an efficient algorithm has been employed. The CVRP (Figure D.4) is a classical routing problem in which a fleet of capacitated vehicles must visit a number of customers and satisfy their demands. The routes start and finish at a specific depot, where all goods are initially held. Moving between nodes (customers or depot) has associated a distance-based cost. Typically, the aim is to minimize the total cost of the designed routes. Despite this simple description, the CVRP is a NP-hard problem with an extraordinary number of real-life applications in the transportation field.

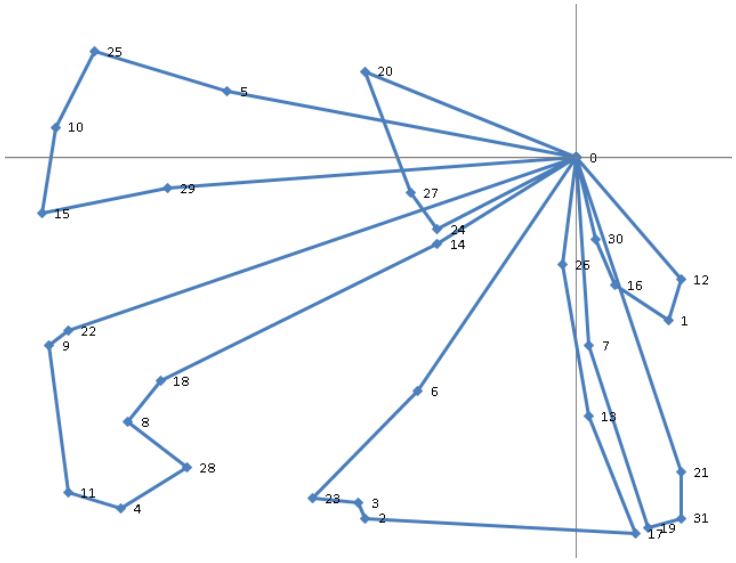


FIGURE D.4: A CVRP instance with customers being served from a central depot.

The randomized version of the popular Clarke and Wright savings (CWS) heuristic (Clarke and Wright, 1964), described in Juan et al. (2014d), has been chosen to solve the CVRP. Initially, this heuristic computes a dummy solution in which there is a route for each customer. Afterwards, a list of all edges connecting two customers is created. It is sorted according to the savings that would be obtained if the dummy solution was modified to include an edge by merging the corresponding routes. This is an iterative process that considers all edges and carries out merges only if no constraint is violated. The randomized version of the heuristic introduces variability in the sort process, erasing the greedy behavior of the classic version. Particularly, it assigns a given probability to each edge to be selected in first place, which is coherent with the savings (i.e., edges with a highest saving will have associated a higher probability). Then, the list and the probabilities are updated, and the process is repeated until all edges have been studied. The Kelly instances are used to test our approach (Golden et al., 2008).

The PFSP (Figure D.5) is a well-known scheduling problem. An instance is characterized by a number of independent jobs that have to be processed on a set of independent machines. Each machine can execute at most one job at a time, and all jobs must be processed in the same order in each machine. The aim is to minimize the maximum completion time, so called makespan.

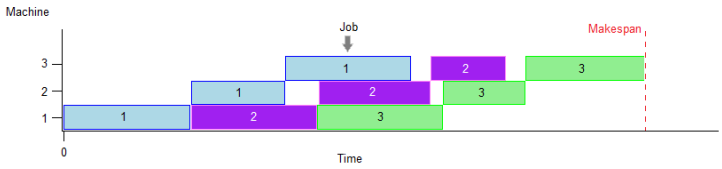


FIGURE D.5: A PFSP instance with 3 machines and 3 jobs.

The ILS-ESP algorithm (Juan et al., 2014a) has been employed to address the PFSP. It relies on the Iterated Local Search (ILS) metaheuristic (Lourenço et al., 2010). In this algorithm, a biased-randomized version of the NEH heuristic (Nawaz et al., 1983) is used to construct an initial solution. The original version computes the total processing time required for each job and creates a solution by iteratively selecting the remaining job with the highest value. The new version is a non-deterministic heuristic capable of generating a set of high-quality solutions without losing the logic behind the original version. Once the initial solution is created, a classical local search is applied which attempts to find a better solution in its neighborhood. The resulting solution, called *baseSol*, is stored. Then, a set of instructions are repeated until a stopping condition (number of iterations or limit of time, for instance) is met. The *baseSol* is perturbed and a local search is applied to the new solution (*newSol*). This solution is stored if it is better than the best solution found so far (*bestSol*). Moreover, the *baseSol* is set to the *newSol* if this solution is better or passes

a Demon-like acceptance criterion (Talbi, 2009). Finally, the best solution found is returned. We have tested our approach on the 120 Taillard's benchmark instances (Taillard, 1993). They are grouped in 12 sets of 10, which are characterized by the following pairs of numbers of jobs and machines: 20x5, 20x10, 20x20, 50x5, 50x10, 50x20, 100x5, 100x10, 100x20, 200x10, 200x20, and 500x20. The instance resolution has been performed considering a specific combination of the parameters 'limit of time' (1, 5, 10, 15, 20, and 30 seconds) and 'number of agents' running in parallel (1, 4, 8, 16, 32, and 64).

All the experiments have been carried out using 64 slaves and 1 master. The master specifications are 3.5GHz Intel Xeon-IvyBridge with 8GB of RAM. The slaves are a heterogeneous set of desktop computers not having more than 8GB of RAM and up to 8 cores each. The machines were connected to the parallel computing environment using one of the following browsers: Microsoft Internet Explorer, Google Chrome or Mozilla Firefox, all of them with JavaScript enabled. The slaves were connected over a usual shared internet connection. For this reason, latencies or high speed connections were considered negligible.

7. Analysis of Results

Figure D.6 shows the results obtained after running the algorithm for solving the Kelly instances during 20 seconds of clock time per instance. Considering all instances, the first boxplot shows the gaps between the best known solution (BKS) and the solution generated by the CWS heuristic. The remaining boxplots show the gaps between the BKS and different executions of our algorithm, each one using a different number of agents running in parallel. The number of agents tested were: 64, 128, and 256. It should be noticed that, for the 20 seconds considered, the distributed approach allows to reduce the gap down to almost 5% even for a reasonably low number of agents (i.e., 64). Note we were using a simple biased-randomized version of the CWS heuristic, this gap could be reduced even further by employing a more powerful algorithm such as the SR-GCWS-CS (Juan et al., 2010). Therefore, the approach can provide reasonably good solutions in just a few seconds without any algorithm fine-tuning or software installation / configuration effort.

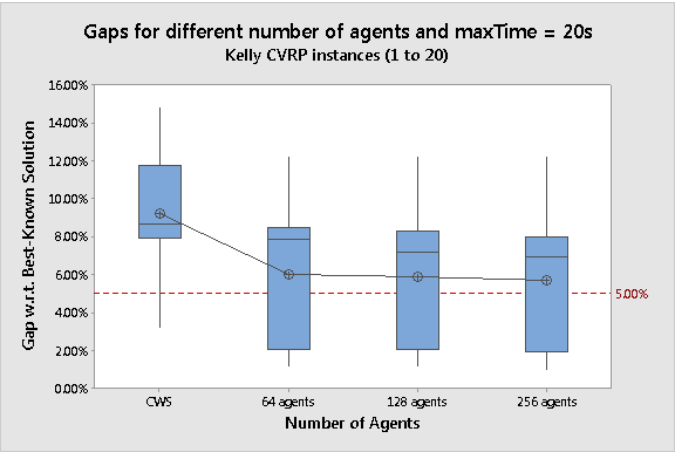


FIGURE D.6: Results for the CVRP using the Kelly instances.

Regarding the PFSP instances, Table D.1 summarizes the results of our computational experiments using a maximum time of 5 seconds. Each row refers to a different set of instances. Each column shows the gap between the BKS and our solution for different numbers of agents (1, 4, 8, 16, 32, and 64). Notice that the gaps shrink as the number of agents working in parallel is increased.

Figure D.7 summarizes similar results for different values of the maximum clock time. It can be observed that, as time increases or as the number of agents increases, the average gap (for the entire set of benchmark instances) decreases. A detailed case is illustrated in Figure D.8, which displays the scatterplot of costs versus limit of time and number of agents for a given instance.

TABLE D.1: Results for the PFSP considering the Taillard instances. Gaps for different number of agents and a maximum time of 5 seconds.

Taillard set	BKS - 1A	BKS - 4A	BKS - 8A	BKS -16A	BKS -32A	BKS - 64A
20x5	0.00%	0.00%	0.00%	0.00%	0.00%	0.00%
20x10	0.08%	0.08%	0.04%	0.00%	0.00%	0.00%
20x20	0.06%	0.02%	0.01%	0.00%	0.00%	0.00%
50x5	0.01%	0.01%	0.01%	0.01%	0.00%	0.00%
50x10	0.84%	0.74%	0.72%	0.65%	0.61%	0.54%
50x20	3.21%	2.85%	2.76%	2.68%	2.65%	2.58%
100x5	0.05%	0.02%	0.00%	0.00%	0.00%	0.00%
100x10	0.53%	0.33%	0.25%	0.22%	0.21%	0.18%
100x20	3.14%	2.67%	2.66%	2.63%	2.56%	2.46%
200x10	0.40%	0.26%	0.24%	0.23%	0.20%	0.14%
200x20	2.36%	2.20%	2.17%	2.15%	2.06%	2.00%
500x20	1.88%	1.53%	1.42%	1.32%	1.32%	1.26%
Averages	1.05%	0.89%	0.86%	0.82%	0.80%	0.76%

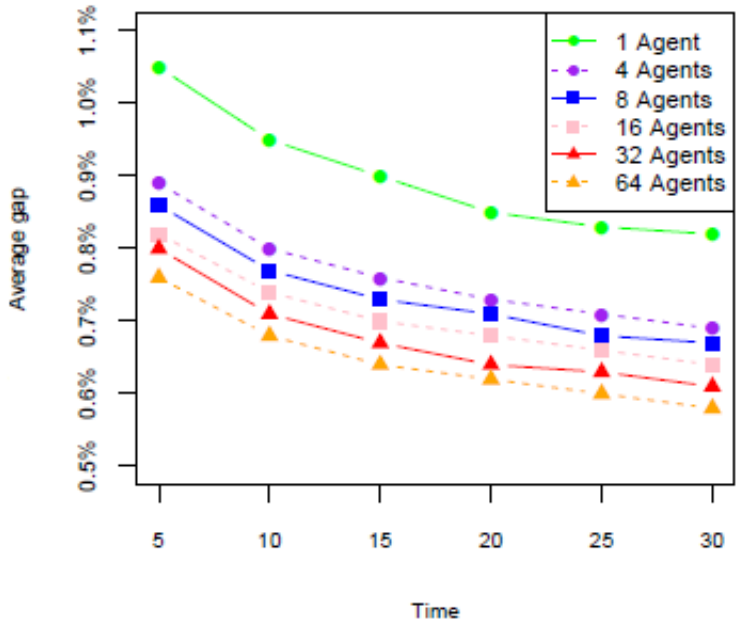


FIGURE D.7: Average gaps for different numbers of agents and limits of time.

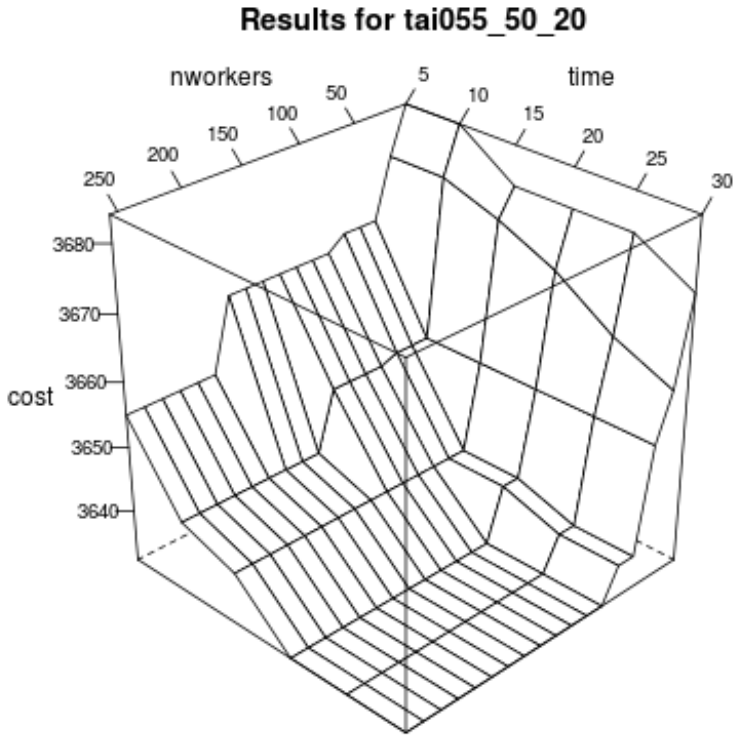


FIGURE D.8: Objective solutions for different numbers of agents and limits of time.

8. Conclusions

This paper has discussed the benefits of Distributed and Parallel Computing Systems (DPCS) for Small and Medium Enterprises (SMEs), which often lack advanced technical skills and modern equipment. In a globalized and dynamic environment, SMEs may become more competitive by obtaining a higher number of computational resources. DPCS offer two possible solutions: the first is to acquire them from an external provider (e.g., a public cloud), and the second consists in seizing the underutilized resources, which is the same aim that characterizes Volunteer Computing but focusing on owned resources. The most relevant advantages are related to data-privacy, costs, and environmental footprint.

We have presented SmartMonkey, a browser-based platform for Distributed Computing. It constitutes a flexible, cheap and simple approach, since it relies on available resources, do not require download/install software, and can be initialized just by accessing a website from a web browser. A JavaScript code embedded in the website allows the communication between the networked resources. Note that this technology is easily scalable due to its simplicity and the possibilities it offers for aggregating new resources.

The potential of DPCS in Mathematical Optimization has already been highlighted in the related literature. Numerous SMEs frequently face complex Combinatorial Optimization Problems (COPs) that require a real-time solution (e.g., flight rescheduling or online routing). Since exact methods are usually incapable of addressing real-sized problem instances in short computing times, approximate methods are widely used. In particular, randomized algorithms may provide a number of high-quality solutions by being repeatedly executed with a different seed. This property suggests an easy way to parallelize the resolution of a problem instance in which each aggregated machine solves the problem instance with a given seed and only the best found solution is returned.

The applicability of the described platform is tested on two classic COPs in the fields of transportation and production. The proposed approach relies on the master-slave architecture and analyzes how solutions vary with time and as the number of available agents is also modified. For the instances analyzed, our approach is able to provide high-quality solutions in real-time.

Acknowledgments

This work has been partially supported by the Spanish Ministry of Economy and Competitiveness (grant TRA2013-48180-C3-P), FEDER, and the Department of Universities, Research & Information Society of the Catalan Government (Grant 2014-CTP-00001). We are also grateful to Incubio for their support during the development of this research work.

References

Anderson, D. P. (2004). "Boinc: a system for public-resource computing and storage". In: *Grid Computing, 2004. Proceedings. Fifth IEEE/ACM International Workshop on*. IEEE, pp. 4–10.

Armbrust, M., A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia (2009). "Above the clouds: A berkeley view of cloud computing". In: *Technical Report UCB/EECS-2009-28, EECS Department, University of California*.

Berry, K. (2009). "Distributed and Grid Computing via the browser". In:

Buyya, R. and S. Venugopal (2005). "A gentle introduction to grid computing and technologies". In: *Database* 2, R3.

Cabrera, G. (2014). "Service allocation methodologies for contributory computing environments". In: *Universitat Oberta de Catalunya*.

Cérin, C. and G. Fedak (2012). *Desktop grid computing*. CRC Press.

Chien, A., B. Calder, S. Elbert, and K. Bhatia (2003). "Entropia: architecture and performance of an enterprise desktop grid system". In: *Journal of Parallel and Distributed Computing* 63.5, pp. 597–610.

Clarke, G. and J. Wright (1964). "Scheduling of vehicles from a central depot to a number of delivering points". In: *Operations Research* 12, pp. 568–581.

Coulouris, G. F., J. Dollimore, and T. Kindberg (2005). *Distributed systems: concepts and design*. Pearson education.

Foster, I. and C. Kesselman (2003). *The Grid 2: Blueprint for a new computing infrastructure*. Elsevier.

Golden, B. L., S. Raghavan, and E. A. Wasil (2008). *The vehicle routing problem: latest advances and new challenges*. Vol. 43. Springer Science & Business Media.

Juan, A. A., J. Faulin, J. Jorba, D. Riera, D. Masip, and B. Barrios (2011a). "On the use of Monte Carlo simulation, cache and splitting techniques to improve the Clarke and Wright savings heuristics". In: *Journal of the Operational Research Society* 62, pp. 1085–1097.

Juan, A. A., J. Faulin, J. Jorba, J. Caceres, and J. M. Marques (2013b). "Using parallel & distributed computing for real-time solving of vehicle routing problems with stochastic demands". In: *Annals of Operations Research* 207.1, pp. 43–65.

Juan, A. A., J. Cáceres-Cruz, S. Gonzalez, D. Riera, and B. Barrios (2014d). "Biased randomization of classical heuristics". In: *Encyclopedia of Business Analytics and Optimization, IGI Global* 1, pp. 314–324.

Juan, A. A., H. R. Lourenço, M. Mateo, R. Luo, and Q. Castella (2014f). "Using iterated local search for solving the flow-shop problem: parallelization, parametrization, and randomization issues". In: *International Transactions in Operational Research* 21.1, pp. 103–126.

Litzkow, M. J., M. Livny, and M. W. Mutka (1988). "Condor-a hunter of idle workstations". In: *Distributed Computing Systems, 1988., 8th International Conference on*. IEEE, pp. 104–111.

Lourenço, H.R., O.C. Martin, and T. Stützle (2010). "Iterated local search: framework and applications". In: *Handbook of Metaheuristics*. Ed. by M. Gendreau and J.-Y. Potvin. Vol. 146. International Series in Operations Research & Management Science. Springer US, pp. 363–397.

Nadiminti, K. and R. Buyya (2005). "Enterprise grid computing: State-of-the-art". In: *Oct-2005. www. cloudbus. org/reports/EOSJArticleTR05. pdf*.

Nawaz, M., J. Enscore, and I. Ham (1983). "A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem". In: *Omega* 11, 91—95.

Sarmenta, L. F. G. (2001). "Volunteer computing". PhD thesis. Massachusetts Institute of Technology.

Taillard, E. (1993). "Benchmarks for basic scheduling problems". In: *European Journal of Operational Research* 64, 278—285.

Talbi, E.-G. (2006). *Parallel combinatorial optimization*. Vol. 58. John Wiley & Sons.
Talbi, E.-G. (2009). *Metaheuristics: From design to implementation*. New Jersey: John Wiley & Sons.

D.2 Combining simulation with metaheuristics in distributed scheduling problems with stochastic processing times

Laura Calvet[1], Victor Fernandez-Viagas[2], Jose M. Framinan[2] and Angel A. Juan[1]

1. Computer Science Department, Open University of Catalonia - IN3
Carl Friedrich Gauss Avenue, Castelldefels 08860, SPAIN
e-mail: {lcalvetl, ajuanp}@uoc.edu
2. School of Engineering, University of Seville
Descubrimientos Avenue, Seville 41092, SPAIN
e-mail: {vfernandezviagas, framinan}@us.es

Abstract

In this paper, we focus on a scenario in which a company or a set of companies conforming a supply network must deliver a complex product (service) composed of several components (tasks) to be processed on a set of parallel flow-shops with a common deadline. Each flow-shop represents the manufacturing of an independent component of the product, or the set of activities of the service. We assume that the processing times are random variables following a given probability distribution. In this scenario, the product (service) is required to be finished by the deadline with a user-specified probability, and the decision-maker must decide about the starting times of each component/task while minimizing one of the following alternative goals: *(a)* the maximum completion time; or *(b)* the accumulated deviations with respect to the deadline. A simheuristic-based methodology is proposed for solving this problem, and a series of computational experiments are performed.

1. Introduction

The complexity of manufacturing has become increasingly higher due to the rise of supply chains where different companies co-operate to manufacture a product for a final customer in a distributed manner. These supply networks naturally appear as companies, in their fierce global competition, try to identify their core competences and outsource/purchase those activities/services in which they do not excel. As a result, these products (or services, as these words will be used interchangeably throughout the paper) can be decomposed into a set of independent components/tasks –each one to be manufactured in a different facility– with a common due date or deadline (Figure D.1). In this paper, we assume that each of these components/tasks has to be processed in a factory, which can be modeled as a permutation flow-shop problem with random or stochastic processing times (PFSPST).

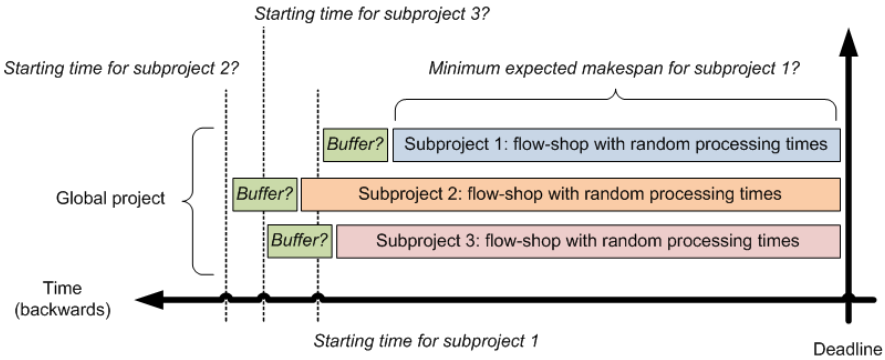


FIGURE D.1: A product requiring 3 independent flow-shops with a common deadline.

In each factory k of a set F (with $k \in \{1, ..., f\}$), a set J_k of n jobs has to be processed by a set M of m machines, being T_{ijk} (abbreviated to T_{ij} when it does not lead to confusion) the random variable representing the time it takes for job i of factory k to be processed by machine j (Figure

D.2). The PFSPST goal is to find a sequence (permutation) of jobs that optimizes a given criterion (taking into account that all jobs are processed by all machines in the order defined by the selected permutation). The most employed criterion in the scientific literature is the minimization of the expected maximum completion time or expected makespan, i.e., the average time it requires to process all the jobs throughout all the machines for the selected permutation.

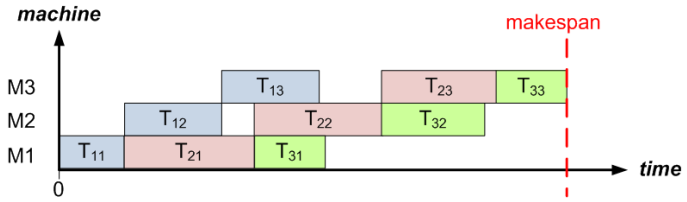


FIGURE D.2: Example of a PFSP with 3 jobs and 3 machines.

Additionally, the global product is required to be finished by the deadline with a user-specified probability, p_0 (from now on, we will assume that the global product is finished if, and only if, all its components/tasks are finished). In this context, the decision-maker must decide about the starting times of each component, while minimizing one of the following alternative goals: *(a)* the maximum completion time of all components (machine-booking goal), i.e., the sum of all expected makespans and buffers in the product; or *(b)* the accumulated deviations with respect to the deadline (synchronization goal).

Our approach for solving this stochastic combinatorial optimization problem is based on the use of a simheuristic algorithm combining simulation (Monte Carlo in this case) with a metaheuristic framework. Some of the main benefits of our approach are the following ones: *(i)* it does not make any assumption on the size of the instances, i.e., being based on a metaheuristic framework it can solve large-scale instances in reasonable computing times; and *(ii)* it does not make any assumption either on the probability distributions employed to model the random processing times, i.e., being based on simulation there is no need to assume normality of processing times –any probability distribution can be used instead.

The rest of the paper is organized as follows: Section 2 provides a review of related work. Section 3 describes the main ideas behind our simulation-optimization approach. Some numerical experiments are carried out in Section 4, and analyzed in Section 5. Finally, we conclude this paper by summarizing its main findings and discussing future work in Section 6.

2. Related work

Different streams of the literature are related to the problem under consideration, namely assembly scheduling, distributed flow-shop scheduling, and permutation flow-shop scheduling with stochastic processing times. These are discussed in the next subsections.

Assembly scheduling

This problem is also denoted *n-stage assembly* or *assembly flow-shop scheduling*. In this problem, m tandem lines are arranged prior to a single assembly station which is fed by the tandem lines. Using this layout, n different products (jobs) have to be manufactured, each one consisting of m components manufactured in the tandem lines. The processing time of each component in each line is different. Some authors distinguish among the *fixed* case (i.e., each component can be processed only in a given tandem line), and the *unfixed* case (i.e., each component can be processed in different factories).

For these problems, different objectives are sought, such as makespan minimization (Sung and Juhn, 2009), total flow-time (Al-Anzi and Allahverdi, 2013; Sung and Kim, 2008), due date fulfillment (Al-Anzi and Allahverdi, 2007), or the combination of several indicators (Seidgar et al., 2014).

Most references refer to the 2-stage case (production followed by assembly), so they assume that each tandem line consists of a single machine. The underlying hypothesis is that there is a single processing time for each component before the assembly process. For this problem, different exact and approximate methods have been proposed, and some variants of the original problem

have been tackled by Sung and Juhn (2009), where two types of components –manufactured and imported– are considered, and by Liao et al. (2015), where assembly batches are assumed.

Several other variants of the problem for three stages have been addressed in the literature (see e.g. Koulamas and Kyparisis (2001)), but in none of the different versions of the problem the processing times have been assumed to be stochastic.

Distributed Flow-shop Scheduling Problem (DFSP)

This problem has several similarities with the one presented in this communication: there are m identical permutation flow-shops where n jobs have to be processed. However, in the DFSP the jobs have not been assigned to each flow-shop, so this assignment becomes part of the decision problem. Furthermore, the DFSP has not been addressed with objectives related with due dates. In fact, the only objective addressed so far refers to makespan minimization (Naderi and Ruiz, 2010), for which the best available heuristic is due to Fernandez-Viagas and Framinan (2015a).

A particular case of the DFSP refers to the so-called *Distributed Assembly Flow-shop Scheduling Problem*, which combines the DFSP with assembly scheduling. In this problem, a distributed flow-shop composed of k identical flow-shops is followed by a single assembly operation. n jobs consisting each one of k components have to be assembled after each component has been manufactured in one of the flow-shops. This decision problem includes job assignment plus the scheduling of jobs in the assembly line. The main references for this problem are Hatami et al. (2015) and Hatami et al. (2013). In the first reference, the authors consider the objective of makespan minimisation, while in the second sequence-dependent setup times are assumed. In both cases, the problem is addressed using approximate algorithms, and, as in the assembly scheduling problems, we are not aware of references dealing with stochastic processing times.

Permutation Flow-shop Problem with Stochastic Processing Times

While the PFSP has been intensively studied during the last few decades, the PFSPST has received less attention. Baker and Trietsch (2011) designed heuristics for addressing the 2-machine PFSPST, where the processing times are independent random variables following specific probability distributions. Later, Baker and Altheimer (2012) presented a methodology for the m-machine version. In addition, several variations of the PFSPST have been analyzed. For instance, Allaoui et al. (2006) and Choi and Wang (2012) worked on the stochastic hybrid flow-shop scheduling problem, aiming to minimize the expected makespan. The same problem was tackled by Kianfar et al. (2012) with the goal of minimizing the average tardiness of jobs. A novel approach is applied in Zhou and Cui (2008) for tackling the multi-objective stochastic PFSP, where both the flow-time and delay time of jobs are minimized.

An interesting line is related to uncertainty. Basically, there are two categories: proactive (or robust) scheduling and reactive scheduling. Works falling in the first category (Roy, 2010) propose constructing an original predictive schedule. The basic aim is to find schedules that do not require new schedules (or significant changes) when confronting disruptions. These works may consider probability distributions or sets of scenarios. Al Kattan and Maragoud (2008), Ghezail et al. (2010) and Liu et al. (2011) addressed the PFSP with uncertainty implementing proactive scheduling strategies. On the other hand, reactive scheduling consists in revising and re-optimizing schedules when unexpected events take place. A classical option is to obtain a predictive scheduling and then try to repair it according to the actual state of the system. A comprehensive review on rescheduling under disruptions is provided by Katragjini et al. (2013).

Some authors employ exact methods for addressing the PFSPST. A disadvantage of many of these methods is that they only work with a specific set of probability distributions and relatively small instances. Moreover, it may be difficult to adapt them for handling dependencies among processing times. Simulation techniques enable researchers to deal with these situations in a natural way. An interesting example is the work of Baker and Altheimer (2012), which proposed a hybrid approach combining heuristics and simulation. The authors tested three heuristic methods: two relying on the CDS heuristic (Campbell et al., 1970) and one on the NEH heuristic (Nawaz et al., 1983).

3. Proposed Methodology

Our methodology is based on a simheuristic approach (Juan et al., 2015a), which relies on the Iterated Local Search (ILS) metaheuristic (Lourenço et al., 2010) and Monte Carlo simulation (MCS) techniques. This metaheuristic has been successfully applied to a wide range of combinatorial optimization problems and is highly popular among researchers. Due to its modularity, it is relatively easy to implement. On the other hand, MCS techniques enable the assessment of solutions in a dynamic environment by taking the following steps: *(1)* simulate a number of scenarios, where each one is created by generating one value per random variable; *(2)* apply a specific solution in each scenario and compute a measure of performance; and *(3)* calculate the average performance.

The methodology is summarized in Algorithm 1 and described next. The inputs are the set F, the deadline (d_0), the probability p_0, and the processing time of each trio of job, machine, and component ($T_{ijk} \forall i \in J, j \in M$ and $k \in F$), which is assumed to follow a specific probability distribution (either theoretical or empirical). Initially, the number of components ($nComponents$) is obtained. Then, a set of instructions are performed for each component:

1. calculate the associated probability (p_{0k}) in order to reach p_0. p_{0k} is computed as $p_0^{1/\text{nComponents}}$ (note we assume that the components are independent);
2. solve the corresponding PFSPST (i.e., get the 'best' permutation);
3. employ MCS techniques to get a sample of makespans (of size r) for the solution obtained and use it to build an empirical cumulative distribution function;
4. identify the makespan below which p_{0k} percent of the observations may be found (i.e., the p_{0k} percentile); and
5. set the starting time as d_0 minus the p_{0k} percentile and store it.

Once all starting times are computed, the procedure returns them. In this work we define 'best' permutation ($bestSol$) as the one minimizing the expected makespan.

Algorithm 1

```
1:  procedure DistributedFlowshopScheduling(F, d_0, p_0, T_ijk)
    d_0: product deadline (common due date for all components)
    p_0: user-specified probability that the product finishes on or before d_0
    T_ijk: random processing time of job i in machine j for component k
2:      nComponents ← getNumberComponents(F)
3:      for each component k in F do
4:          p_0k ← calcProbabilityComponent(p_0, k, nComponents)
5:          bestSol(k) ← solvePFSPST(k, T_ijk)
                                          ▷ use a simulation-optimization algorithm
6:          distFunction(k) ← calcDistFuntion(bestSol(k))
                                          ▷ use observations from simulation
7:          percentile(p_0k) ← calcPercentil(distFunction(k), p_0k)
8:          startingTime(k) ← d_0 − percentil(p_0k)
9:          startingTimes ← add(startingTime(k))
10:     end for
11:     return startingTimes
12: end procedure
```

Regarding steps 3 and 4, we will provide more details. There is a sample of makespans $x_1, x_2, ..., x_r$, which are observations of independent and identically distributed real random variables with a common cumulative distribution function $F(t)$, which is unknown. The empirical cumulative distribution function is defined as:

$$\hat{F}_r(t) = \frac{\text{number of elements in the sample } \leq t}{r} = \frac{1}{r}\sum_{i=1}^{r} 1_{x_i \leq t} \tag{D.1}$$

By the strong law of large numbers, the estimator $\hat{F}_r(t)$ converges to $F(t)$ as almost surely, for every value of t. As a consequence, the estimator $\hat{F}_r(t)$ is consistent. Thus, we can obtain the

value t ensuring that the component k will have a makespan of t or lower with a probability of p_{0k} by computing: $t = \hat{F}_r^{-1}(p_{0k})$.

In order to solve each of the PFSPST (i.e., step 4), we present a simheuristic methodology similar to the one described in Juan et al. (2014a). It relies on two components: an ILS metaheuristic that searches promising solutions, and MCS techniques for assessing them. Figure D.3 shows the basic scheme. The first steps are to transform the original problem instance into a PFSP instance replacing random variables by their means, and apply the aforementioned NEH heuristic for finding a solution (*baseSol*). Then, MCS techniques are used to compute the expected makespan for that solution considering the stochastic environment described by the original instance. A new solution (*bestSol*) is created to store the best solution found, which is a copy of *baseSol* at this stage. Afterwards, the loop is started, which will execute instructions during *maxTime* seconds. The first step is building a new solution (*newSol*) by perturbing *baseSol*, which implies selecting randomly two jobs and interchanging their positions. A local search based on the classical shift-to-left movement is applied to each job in a random order. The next step checks whether the makespan of *newSol* is lower than or equal to that of *baseSol*. If this is not satisfied, the solution is discarded and another iteration starts. Otherwise, the expected makespan (obtained using MCS techniques) of *newSol* is computed. A variable *delta* contains the difference between the expected makespan of *newSol* and *baseSol*. If *delta* is negative or 0 (i.e., there is an improvement) *newSol* replaces *baseSol* and the variable *credit* (which is initially 0) is reset to $-delta$. Additionally, *bestSol* is updated if its expected makespan is higher than the one of *baseSol*. If *delta* is positive but equal to or lower than *credit* (i.e., it is only 'slightly' worse), *baseSol* is updated and *credit* is reset to 0. This acceptance criterion that chooses whether *baseSol* is updated is known as Demon-like process (Talbi, 2009) and is employed to help avoiding local minima during the execution of the algorithm. Finally, *bestSol* is returned. It is important to note the difference between the methodology described in Juan et al. (2014a) and ours: while the former is deterministic-driven (i.e., the replacement of the base solution depends on the makespan of the transformed instance), the latter is stochastic-driven (i.e., stochastic makespans are employed). This modification provides better solutions without requiring more computational time.

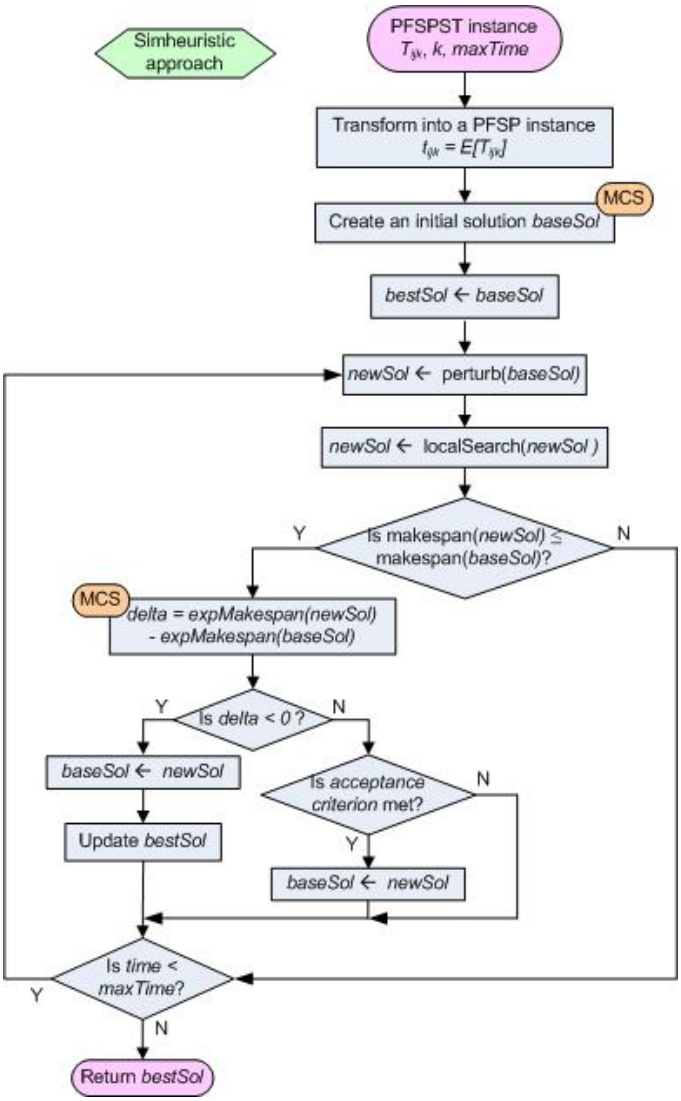


FIGURE D.3: Flowchart of the proposed methodology for the PFSPST.

4. Computational experiments

The methodology presented has been implemented as a Java application. A standard personal computer, Intel Core i5 CPU at 3.2 GHz and 4 GB RAM with Windows 7 has been employed. We have experimented with 4 instances with the parameters $f \in \{2, 4\}$ and $m \in \{5, 10\}$, and with 20 jobs assigned to each line $k \in F$. The processing times on each machine are taken from the PFSP instances introduced in Taillard (1993). Table D.1 describes the composition of the new instances and their main characteristics (d_0, m, and f) in the third, fourth and fifth columns, respectively. Second column indicates the instance of Taillard (1993) from which the processing times are considered. The common due date has been generated by doing $d_0 = h \cdot C$, following a similar procedure to the ones described by Della Croce et al. (2000) and Biskup and Feldmann (2001), where h is a parameter to indicate how loose/tight the common due date is and C is the makespan of the line, i.e., the highest makespan among all the factories. Note that the makespan of each line is the best known solution of the corresponding Taillard instance which is recorded in http://mistic.heig-vd.ch/. To avoid unfeasible instances, we generate loose common due dates according to a uniform distribution [1.2, 1.6] for the parameter h. All experiments have been run 5 times with different seeds and only the best values are stored. The computational time allowed for each instance is the sum of the time assigned to each production line, which is $0.03 \cdot m \cdot n$ seconds (as suggested in Juan et al. (2014a)). For introducing the stochasticity of processing times, each one is defined as an independent random variable following a logNormal

Algorithm 2

```
procedure SOLVEPFSPST(k, maxTime)
k: subproject with a specific stochastic flow-shop associated, which is characterized by a
random processing time T_ijk for each job i and machine j
maxTime: maximum time for the loop inside the ILS
    t_ij ← E[T_ijk]
    baseSol ← calcDetMakespan(t_ij)                          ▷ use the NEH heuristic
    baseSol ← calcStochMakespan(baseSol)                     ▷ use simulation
    bestSol ← baseSol
    while time ≤ maxTime do
        newSol ← perturb(baseSol)
        newSol ← localSearch(newSol)
        if detMakespan(newSol) ≤ detMakespan(baseSol) then   ▷ newSol is promising
            newSol ← calcStochMakespan(newSol)               ▷ use simulation
            delta ← stochMakespan(newSol) - stochMakespan(baseSol)
            if delta ≤ 0 then                                ▷ newSol improves baseSol
                baseSol ← newSol
                credit ← −delta
                if stochMakespan(newSol) ≤ stochMakespan(bestSol) then   ▷ newSol
improves bestSol
                    bestSol ← newSol
                end if
            else if delta ≤ credit then                      ▷ acceptance criterion
                baseSol ← newSol
                credit ← 0
            end if
        end if
    end while
    return bestSol
end procedure
```

distribution with a mean (μ) set to the processing time of the original instance and a standard deviation (σ) obtained from the coefficient of variation ($c = \sigma/\mu$). Three values of c are tested: 0.1, 0.5, and 1 (proposed in Framinan and Perez-Gonzalez (2015)), which represent a low, medium, and high level of stochasticity, respectively. Regarding the user-given probabilities, three levels are established: 0.8, 0.9, and 0.95.

First, we compare the performance of our algorithm following a stochastic approach (the described in the previous section) with a deterministic one, which assumes zero variability. Table D.2 displays the booking times (i.e., the sum of all expected makespans and buffers, or percentiles) for the first instance considering each one of the 9 scenarios defined by the probabilities and the levels of stochasticity. The mean gaps between the booking times of both approaches for each level of stochasticity (from low to high) are: -0.12%, -1.13%, and -1.77%, respectively. Similarly, for each level of probability (from low to high), the mean gaps are: -0.89%, -0.99%, and -1.16%, respectively. Secondly, we analyze the booking times and buffers for the same 9 scenarios studying all instances. Solutions are provided in Table D.3. Figure D.4 shows this information in boxplots. Finally, the relation between booking times and level of stochasticity is studied for a high number of probability values. The patterns identified for the first instance are presented in Figure D.5.

TABLE D.1: Description of the instances.

Inst.	Taillard's instances	d_0	m	f
ins1	ta001 ($f = 1$), ta002 ($f = 2$)	1841	5	2
ins2	ta003 ($f = 1$), ta004 ($f = 2$), ta005 ($f = 3$), ta006 ($f = 4$)	1620	5	4
ins3	ta011 ($f = 1$), ta012 ($f = 2$)	2409	10	2
ins4	ta013 ($f = 1$), ta014 ($f = 2$), ta015 ($f = 3$), ta016 ($f = 4$)	2372	10	4

TABLE D.2: Booking times for "ins1" following the stochastic and the deterministic approach.

		Stochastic approach			Deterministic approach			
	Stoch.	Low	Medium	High	Low	Medium	High	Average
Prob.	0.8	2671.96	2845.83	3086.30	2674.66	2876.29	3133.19	2881.37
	0.9	2688.03	2944.64	3327.98	2691.21	2977.94	3386.28	3002.68
	0.95	2701.60	3034.82	3571.91	2705.70	3072.52	3648.68	3122.54
	Average	2687.19	2941.76	3328.73	2690.52	2975.58	3389.38	

TABLE D.3: Table of results.

		Level of stochasticity					
		Low		Medium		High	
		Booking time	Buffer	Booking time	Buffer	Booking time	Buffer
ins1	0.8	2671.96	24.95	2845.83	104.23	3086.30	204.88
	0.9	2688.03	41.02	2944.64	203.04	3327.98	446.55
	0.95	2701.60	54.59	3034.82	293.22	3571.91	690.49
ins2	0.8	4867.96	38.18	5287.05	215.53	5821.46	445.30
	0.9	4896.62	66.85	5515.96	444.43	6346.50	970.34
	0.95	4923.97	94.20	5736.55	665.02	6879.40	1503.25
ins3	0.8	3270.41	13.66	3478.39	98.77	3786.69	221.03
	0.9	3282.55	25.80	3579.86	200.24	4041.78	476.12
	0.95	3293.67	36.91	3680.14	300.53	4308.61	742.95
ins4	0.8	5748.58	29.41	6069.68	163.80	6544.70	358.62
	0.9	5774.13	54.96	6251.65	345.78	7012.24	826.16
	0.95	5797.32	78.15	6427.30	521.42	7485.07	1298.99

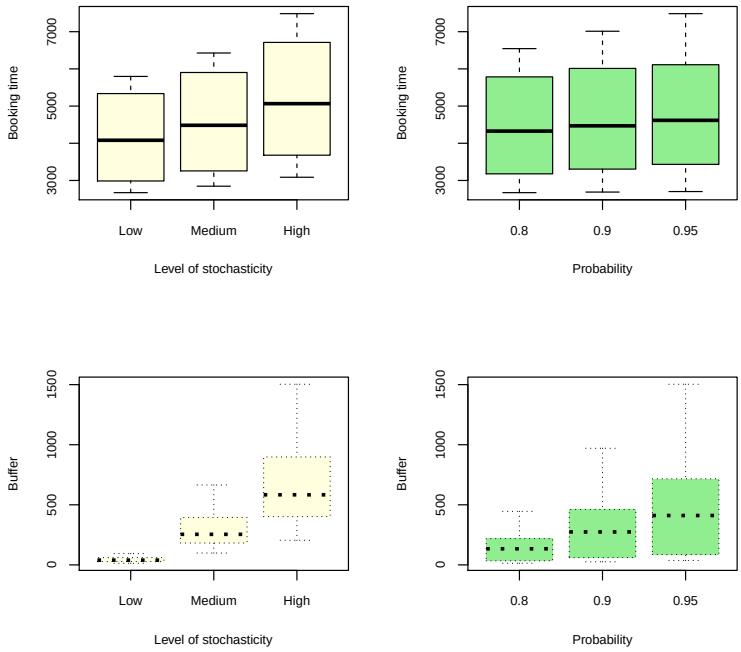


FIGURE D.4: Boxplots of booking times and buffers for several scenarios.

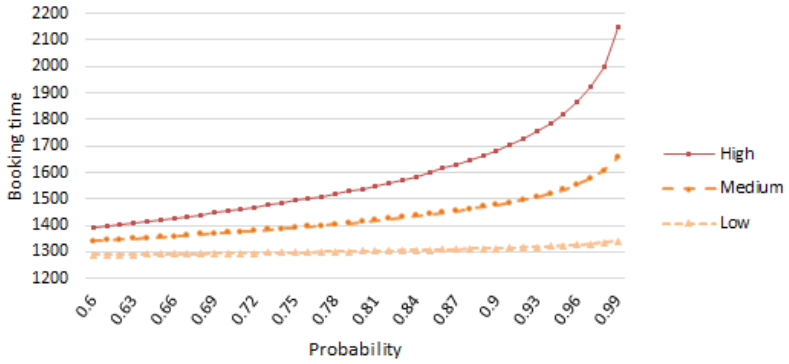


FIGURE D.5: Relation between probability, booking time and level of stochasticity for "ins1".

5. Analysis of results

It is important to start with the comparison between the stochastic and the deterministic approach, in order to check whether introducing simulation in the solving methodology for the PFSPST is useful. The results (Table D.2) indicate that the improvement in terms of booking times is highly significant, with gaps increasing with the levels of stochasticity and probability.

Focusing on the effect of these variables (stochasticity and probability) on booking times, it can be observed (Table D.3 and Figure D.5) that the booking times increase with both the stochasticity and the probability. While the differences among levels of stochasticity are relatively small for low probabilities (below 0.8), they rapidly increase for higher values. As expected, when users require a high probability, the booking time increases exponentially for all the levels of stochasticity considered.

According to the graphical results in Figure D.4, which refer to all instances, both variables have a positive effect on the booking time, being the level of variability more relevant. Regarding the buffers required, they are much more volatile. For a low level of stochasticity, it is very small

but rapidly increases as the level grows. The effect of the probability is a bit smaller but also positive and growing.

6. Conclusions and further research

In this paper we have discussed how simulation can be combined with metaheuristics in order to deal with stochastic multi-factory scheduling problems. In particular, we have studied a scheduling problem composed of parallel and independent components/tasks with a common due date, the processing of each of these components being modeled as a permutation flow-shop problem with stochastic processing times. In this context, a natural question arises: how to set starting times of each component in such a way that the total machine-occupancy time is minimized while ensuring a user-given probability of finishing all components in due time. The computational experiments carried out in this paper show how starting times vary according to factors such as the variance level in the random processing times or the user-required probability threshold for the product to finish on time. Other similar questions can be formulated, e.g., how to set starting times of each component in order to minimize total deviations from the common due date while respecting the aforementioned constraint, etc. These are open research lines that must require from similar approaches to the one introduced here.

Acknowledgments

This work has been partially supported by the Spanish Ministry of Economy and Competitiveness (grants TRA2013-48180-C3-P, DPI2013-44461-P), FEDER, and the Catalan Government (2014-CTP-00001).

References

Al-Anzi, F. and A. Allahverdi (2007). "A self-adaptive differential evolution heuristic for two-stage assembly scheduling problem to minimize maximum lateness with setup times". In: *European Journal of Operational Research* 182.1, pp. 80–94.

Al-Anzi, F. and A. Allahverdi (2013). "An artificial immune system heuristic for two-stage multi-machine assembly scheduling problem to minimize total completion time". In: *Journal of Manufacturing Systems* 32.4, pp. 825–830.

Al Kattan, I. and R. Maragoud (2008). "Performance analysis of flowshop scheduling using genetic algorithm enhanced with simulation". In: *International Journal of Industrial Engineering: Theory, Applications and Practice* 15.1, pp. 62–72.

Allaoui, H., S. Lamouri, and M. Lebbar (2006). "A robustness framework for a stochastic hybrid flow shop to minimize the makespan". In: *International Conference on Service Systems and Service Management*. Vol. 2. IEEE, pp. 1097–1102.

Baker, K. R. and D. Altheimer (2012). "Heuristic solution methods for the stochastic flow shop problem". In: *European Journal of Operational Research* 216.1, pp. 172–177.

Baker, K. R. and D. Trietsch (2009). "Safe scheduling: setting due dates in single-machine problems". In: *European Journal of Operational Research* 196.1, pp. 69–77.

— (2011). "Three heuristic procedures for the stochastic, two-machine flow shop problem". In: *Journal of Scheduling* 14.5, pp. 445–454.

Biskup, D. and M. Feldmann (2001). "Benchmarks for scheduling on a single machine against restrictive and unrestrictive common due dates". In: *Computers & Operations Research* 28.8, pp. 787–801.

Campbell, H. G., R. A. Dudek, and M. L. Smith (1970). "A heuristic algorithm for the n job, m machine sequencing problem". In: *Management science* 16.10, B630–B637.

Choi, S. H. and K. Wang (2012). "Flexible flow shop scheduling with stochastic processing times: a decomposition-based approach". In: *Computers & Industrial Engineering* 63.2, pp. 362–373.

Della Croce, F., J. N. D. Gupta, and R. Tadei (2000). "Minimizing tardy jobs in a flowshop with common due date". In: *European Journal of Operational Research* 120.2, pp. 375–381.

Fernandez-Viagas, V. and J. M. Framinan (2015a). "A bounded-search iterated greedy algorithm for the distributed permutation flowshop scheduling problem". In: *International Journal of Production Research* 53.4, pp. 1111–1123.

— (2015c). "NEH-based heuristics for the permutation flowshop scheduling problem to minimise total tardiness". In: *Computers & Operations Research* 60, 27—36.

Framinan, J. M. and P. Perez-Gonzalez (2015). "On heuristic solutions for the stochastic flowshop scheduling problem". In: *Journal of Operational Research* 246.2, 413—420.

Ghezail, F., H. Pierreval, and S. Hajri-Gabouj (2010). "Analysis of robustness in proactive scheduling: a graphical approach". In: *Computers & Industrial Engineering* 58.2, pp. 193–198.

Hatami, S., R. Ruiz, and C. Andrés-Romano (2013). "The distributed assembly permutation flowshop scheduling problem". In: *International Journal of Production Research* 51.17, pp. 5292–5308.

— (2015). "Heuristics and metaheuristics for the distributed assembly permutation flowshop scheduling problem with sequence dependent setup times". In: *International Journal of Production Economics* 169, pp. 76–88.

Juan, A. A., J. Faulin, J. Jorba, D. Riera, D. Masip, and B. Barrios (2011a). "On the use of Monte Carlo simulation, cache and splitting techniques to improve the Clarke and Wright savings heuristics". In: *Journal of the Operational Research Society* 62, pp. 1085–1097.

Juan, A. A., B. Barrios, E. Vallada, D. Riera, and J. Jorba (2014a). "A simheuristic algorithm for solving the permutation flow shop problem with stochastic processing times". In: *Simulation Modelling Practice and Theory* 46, pp. 101–117.

Juan, A. A., J. Faulin, S. Grasman, M. Rabe, and G. Figueira (2015a). "A review of simheuristics: extending metaheuristics to deal with stochastic optimization problems". In: *Operations Research Perspectives* 2, pp. 62–72.

Katragjini, K., E. Vallada, and R. Ruiz (2013). "Flow shop rescheduling under different types of disruption". In: *International Journal of Production Research* 51.3, pp. 780–797.

Kianfar, K., S. M. T. F. Ghomi, and A. O. Jadid (2012). "Study of stochastic sequence-dependent flexible flow shop via developing a dispatching rule and a hybrid GA". In: *Engineering Applications of Artificial Intelligence* 25.3, pp. 494–506.

Koulamas, C. and G. J. Kyparisis (2001). "The three-stage assembly flowshop scheduling problem". In: *Computers & Operations Research* 28.7, pp. 689–704.

Liao, C.-J., C.-H. Lee, and H.-C. Lee (2015). "An efficient heuristic for a two-stage assembly scheduling problem with batch setup times to minimize makespan". In: *Computers & Industrial Engineering* 88, pp. 317–325.

Liu, Q., S. Ullah, and C. Zhang (2011). "An improved genetic algorithm for robust permutation flowshop scheduling". In: *The International Journal of Advanced Manufacturing Technology* 56.1-4, pp. 345–354.

Lourenço, H.R., O.C. Martin, and T. Stützle (2010). "Iterated local search: framework and applications". In: *Handbook of Metaheuristics*. Ed. by M. Gendreau and J.-Y. Potvin. Vol. 146. International Series in Operations Research & Management Science. Springer US, pp. 363–397.

Naderi, B. and R. Ruiz (2010). "The distributed permutation flowshop scheduling problem". In: *Computers & Operations Research* 37.4, pp. 754–768.

Nawaz, M., J. Enscore, and I. Ham (1983). "A heuristic algorithm for the m-machine, n-job flowshop sequencing problem". In: *Omega* 11, 91—95.

Roy, B. (2010). "Robustness in operational research and decision aiding: a multi-faceted issue". In: *European Journal of Operational Research* 200.3, pp. 629–638.

Seidgar, H., M. Kiani, M. Abedi, and H. Fazlollahtabar (2014). "An efficient imperialist competitive algorithm for scheduling in the two-stage assembly flow shop problem". In: *International Journal of Production Research* 52.4, pp. 1240–1256.

Sung, C. S. and J. Juhn (2009). "Makespan minimization for a 2-stage assembly scheduling problem subject to component available time constraint". In: *International Journal of Production Economics* 119.2, pp. 392–401.

Sung, C. S. and H. A. Kim (2008). "A two-stage multiple-machine assembly scheduling problem for minimizing sum of completion times". In: *International Journal of Production Economics* 113.2, pp. 1038–1048.

Taillard, E. (1993). "Benchmarks for basic scheduling problems". In: *European Journal of Operational Research* 64, 278—285.

Talbi, E.-G. (2009). *Metaheuristics: From design to implementation*. New Jersey: John Wiley & Sons.

Zhou, Q. and X. Cui (2008). "Research on multiobjective flow shop scheduling with stochastic processing times and machine breakdowns". In: *IEEE International Conference on Service Operations and Logistics, and Informatics*. Vol. 2. IEEE, pp. 1718–1724.