

UNIVERSITAT OBERTA DE CATALUNYA
INTERNET INTERDISCIPLINARY INSTITUTE
FACULTY OF COMPUTER SCIENCE, MULTIMEDIA AND TELECOMMUNICATIONS

DOCTORAL THESIS
DOCTORAL PROGRAMME IN NETWORK & INFORMATION TECHNOLOGIES



*A thesis submitted in fulfillment of the requirements for the degree of
Doctor of Philosophy in Network & Information Technologies*

Applications of Biased-Randomized Algorithms and Simheuristics in Integrated Logistics

Author:

Leandro do Carmo Martins

Thesis Committee Members:

Prof. Dr. Angel A. Juan Perez

Prof. Dr. Helena R. Lourenço

Prof. Dr. Marccone J. F. Souza

Barcelona, Spain

September 13, 2021

"It's in making decisions that we learn to decide."

Paulo Freire, *Pedagogy of Freedom*

Acknowledgements

It is time to wrap one more chapter of my professional life. First of all, I would like to thank **God** for being with me whenever I needed Him. I never imagined getting this far.

To **Prof. Dr. Angel A. Juan**, I would like to deeply thank you for having supervised and believed in me during these past four years. You taught me a lot, and your knowledge is an example to me. You encouraged me to come to Barcelona, to live in Vienna for a few months, and you have wonderfully fulfilled your role as a director and supervisor of this important professional step. Thank you very much.

To **Prof. Dr. Helena Ramalhinho**, thanks for introducing and presenting me the right possibilities to make this opportunity feasible to happen. You were the first person I had contact with back in 2016, and thanks to the support of both you and Prof. Dr. Juan, this journey was possible to happen the following year. I will always be grateful for that.

To **Prof. Dr. Patrick Hirsch**, I want to thank you for kindly receiving and supervising me at the University of Natural Resources and Life Sciences (BOKU), in Vienna, Austria. This research stay was very valuable for both my professional and personal life.

To **Prof. Dr. Marcene J. F. Souza**, I would like to thank you for supporting me whenever it was needed.

To my friends from the ICSO group and IN3 building, especially **Sara, Javier, Rafael, Pedro, Christopher, Julio, Alejandro, Laura, Juliana, Stephanie, Maria, Cristina, Maria del Carmen, Santiago, and Behnam**, I want to thank you for all the support you gave me and the great moments we spent together.

To the friends that I met in Barcelona, especially **Brenda, Marcelus, Iara, Vanessa, Tayrine, Vera, Lucas, Thyago, and Alex**, you made this journey a happier one.

To my closest friends from Brazil, especially **Rafael, Samara, Ana, Marina, Mariana, Priscila, and Laís**, you are part of this chapter too.

À minha família, especialmente à minha mãe **Rosa**, minha tia **Carmen**, minha irmã **Renatha**, e minha prima **Marcela**, gostaria de dizer que o apoio de vocês foi fundamental para seguir em frente e fazer o meu melhor. Todo meu esforço é por vocês.

Moreover, I would like to thank **myself** for persisting and not giving up. Those were years of many challenges, struggles, and questions. These were also years of grand discoveries, achievements, and self-knowledge.

Finally, I thank **Barcelona** for providing me some of the best years of my life.

UNIVERSITAT OBERTA DE CATALUNYA
INTERNET INTERDISCIPLINARY INSTITUTE
FACULTY OF COMPUTER SCIENCE, MULTIMEDIA AND TELECOMMUNICATIONS

DOCTORAL PROGRAMME IN NETWORK & INFORMATION TECHNOLOGIES

Abstract

Applications of Biased-Randomized Algorithms and Simheuristics in Integrated Logistics

Transportation and logistics (T&L) activities play a vital role in the development of many businesses from different industries, and the so-called smart sustainable cities. With the increasing number of people living in urban, metropolitan, and peri-urban areas, and the expansion of the on-demand economy and the e-commerce activities, the number of services from transportation and delivery of products that occur in urban areas has considerably increased. Consequently, several urban problems have been potentialized, such as traffic congestion and pollution, since the urban infrastructure is directly used. Urban logistics, therefore, became a critical decision in which many companies and logistics service providers from different industries are willing to improve the efficiency of their operations. Improving the efficiency of logistics activities holds not only the capability of reducing costs for providers but also of providing customers with better services and lower prices. Several related problems can be formulated as a combinatorial optimization problem (COP). Since most of them are *NP-Hard*, because of their combinatorial nature, the finding of optimal solutions through exact solution methods is often impractical in a reasonable amount of time, especially for large-size instances that need high-quality solutions in relatively short computational times. In realistic settings, where problems are even more complex, constrained, and dynamic, the increasing need for ‘instant’ or ‘fast’ decision-making further refutes the use of exacts approaches in real life. Under these circumstances, this thesis aims at: (i) identifying realistic COPs from different industries; (ii) developing different classes of approximate solution approaches (heuristics, biased-randomized heuristics, metaheuristics, simheuristics, and agile optimization) to solve these inherent T&L problems of smart and sustainable cities; (iii) conducting a series of computational experiments in order to validate and measure the performance of the developed approaches; and (iv) writing down conclusions regarding their use on the solving of the identified problems. The novel concept of ‘agile optimization’ is introduced, which refers to the combination of biased-randomized heuristics with parallel computing to deal with real-time decision-making. Apart from solving these practical problems, a survey on the challenges in optimization for ride-sharing –automobile-based shared transportation– systems is presented. Challenges and future lines of research are addressed, specifically in the context of telecommunication and ride-sharing systems, where immediate decision-making requires the employment of agile optimization strategies to provide efficient solutions in real-time.

UNIVERSITAT OBERTA DE CATALUNYA
INTERNET INTERDISCIPLINARY INSTITUTE
FACULTY OF COMPUTER SCIENCE, MULTIMEDIA AND TELECOMMUNICATIONS

DOCTORAL PROGRAMME IN NETWORK & INFORMATION TECHNOLOGIES

Resumen

Applications of Biased-Randomized Algorithms and Simheuristics in Integrated Logistics

Las actividades de transporte y logística (T&L) juegan un papel vital en el desarrollo de muchas empresas de diferentes industrias y las llamadas ciudades inteligentes y sostenibles. Con el creciente número de personas que viven en áreas urbanas, metropolitanas y periurbanas, y la expansión de la economía bajo demanda y las actividades de comercio electrónico, la cantidad de servicios de transporte y entrega de productos que se producen en áreas urbanas ha aumentado considerablemente. En consecuencia, se han potencializado varios problemas urbanos, como la congestión del tráfico y la contaminación, ya que se utiliza directamente la infraestructura urbana. La logística urbana, por tanto, se convirtió en una decisión crítica en la que muchas empresas y proveedores de servicios logísticos de diferentes industrias están dispuestos a mejorar la eficiencia de sus operaciones. Mejorar la eficiencia de las actividades logísticas no solo tiene la capacidad de reducir los costos para los proveedores, sino también de brindar a los clientes mejores servicios y precios más bajos. Varios problemas relacionados pueden formularse como un problema de optimización combinatoria (COP). Dado que la mayoría de ellos son *NP-Hard*, debido a su naturaleza combinatoria, la búsqueda de soluciones óptimas a través de métodos de solución exactos a menudo no es práctico en un período de tiempo razonable, especialmente para instancias de gran tamaño que necesitan soluciones de alta calidad en tiempos computacionales relativamente cortos. En entornos realistas, donde los problemas son aún más complejos, limitados y dinámicos, la creciente necesidad de una toma de decisiones “instantánea” o “rápida” refuta aún más el uso de enfoques exactos en la vida real. En estas circunstancias, esta tesis tiene como objetivo: (i) identificar COP realistas de diferentes industrias; (ii) desarrollar diferentes clases de enfoques de solución aproximada (heurística, heurística sesgada-aleatorizada, metaheurística, simheurística y optimización ágil) para resolver estos problemas de T&L inherentes a las ciudades inteligentes y sostenibles; (iii) realizar una serie de experimentos computacionales para validar y medir el desempeño de los enfoques desarrollados; y (iv) redactar conclusiones iniciales sobre su uso en la resolución de los problemas identificados. Se introduce el novedoso concepto de optimización ágil (*agile optimization*), que se refiere a la combinación de heurísticas sesgadas y aleatorias con computación paralela para hacer frente a la toma de decisiones en tiempo real. Además de resolver estos problemas prácticos, se presenta una encuesta sobre los desafíos en la optimización de los sistemas de viajes compartidos –transporte compartido basado en automóviles. Se abordan desafíos y líneas de investigación futuras, específicamente en el contexto de los sistemas de telecomunicaciones y viajes compartidos, donde la toma de decisiones inmediata requiere el empleo de estrategias ágiles de optimización para brindar soluciones eficientes en tiempo real.

UNIVERSITAT OBERTA DE CATALUNYA
INTERNET INTERDISCIPLINARY INSTITUTE
FACULTY OF COMPUTER SCIENCE, MULTIMEDIA AND TELECOMMUNICATIONS

DOCTORAL PROGRAMME IN NETWORK & INFORMATION TECHNOLOGIES

Resum

Applications of Biased-Randomized Algorithms and Simheuristics in Integrated Logistics

Les activitats de transport i logística (T&L) juguen un paper vital en el desenvolupament de moltes empreses de diferents indústries i de les anomenades ciutats sostenibles intel·ligents. Amb l'augment del nombre de persones que viuen a les zones urbanes, metropolitanes i periurbanes, i l'expansió de l'economia a la carta i de les activitats de comerç electrònic, el nombre de serveis de transport i lliurament de productes que es produeix a les zones urbanes ha tingut augmentat considerablement. En conseqüència, s'han potencialitzat diversos problemes urbans, com la congestió del trànsit i la contaminació, ja que la infraestructura urbana s'utilitza directament. La logística urbana, per tant, es va convertir en una decisió crítica en què moltes empreses i proveïdors de serveis logístics de diferents indústries estan disposats a millorar l'eficiència de les seves operacions. La millora de l'eficiència de les activitats logístiques manté no només la capacitat de reduir costos per als proveïdors, sinó també de proporcionar als clients millors serveis i preus més baixos. Es poden formular diversos problemes relacionats com a problema d'optimització combinatòria (COP). Atès que la majoria d'ells són *NP-Hard*, a causa de la seva naturalesa combinatòria, la cerca de solucions òptimes mitjançant mètodes de solució exactes sovint no és pràctica en un temps raonable, especialment per a instàncies de grans dimensions que necessiten solucions d'alta qualitat en temps computacionals relativament curts. En entorns realistes, on els problemes són encara més complexos, restringits i dinàmics, la creixent necessitat de prendre decisions "instantànies" o "ràpides" refuta encara més l'ús d'enfocaments exactes a la vida real. En aquestes circumstàncies, aquesta tesi té com a objectiu: (i) identificar COP realistes de diferents indústries; (ii) desenvolupant diferents classes d'aproximacions aproximades a la solució (heurística, heurística aleatoritzada, metaheurística, simheurística i optimització àgil) per resoldre aquests problemes inherents de T&L de ciutats intel·ligents i sostenibles; (iii) realitzant una sèrie d'experiments computacionals per tal de validar i mesurar el rendiment dels enfocaments desenvolupats; i (iv) escrivint clares conclusions sobre el seu ús en la resolució dels problemes identificats. S'introdueix el nou concepte d'optimització àgil (*agile optimization*), que fa referència a la combinació d'heurístiques parcialitzades i aleatòries amb informàtica paral·lela per fer front a la presa de decisions en temps real. A part de resoldre aquests problemes pràctics, es presenta una enquesta sobre els desafiaments de l'optimització de sistemes de transport compartit (transport compartit basat en automòbils). Els reptes i les futures línies d'investigació s'aborden, específicament en el context de sistemes de telecomunicacions i compartició de viatges, on la presa de decisions immediata requereix l'ús d'estratègies d'optimització àgils per proporcionar solucions eficients en temps real.

UNIVERSITAT OBERTA DE CATALUNYA
INTERNET INTERDISCIPLINARY INSTITUTE
FACULTY OF COMPUTER SCIENCE, MULTIMEDIA AND TELECOMMUNICATIONS

DOCTORAL PROGRAMME IN NETWORK & INFORMATION TECHNOLOGIES

Resumo

Applications of Biased-Randomized Algorithms and Simheuristics in Integrated Logistics

As atividades de transporte e logística (T&L) desempenham um papel vital no desenvolvimento de muitas empresas de diferentes setores e das chamadas cidades sustentáveis inteligentes. Com o aumento do número de pessoas vivendo em áreas urbanas, metropolitanas e periurbanas, e a expansão da economia sob demanda e das atividades de comércio eletrônico, o número de serviços de transporte e entrega de produtos que ocorre em áreas urbanas aumentou consideravelmente. Consequentemente, diversos problemas urbanos tem sido potencializados, como congestionamento de tráfego e poluição, uma vez que a infraestrutura urbana é diretamente utilizada. A logística urbana, portanto, tornou-se uma decisão crítica, na qual muitas empresas e prestadores de serviços logísticos de diferentes setores estão dispostos a melhorar a eficiência de suas operações. Melhorar a eficiência das atividades logísticas traz não apenas a capacidade de reduzir custos para os fornecedores, mas também de oferecer aos clientes melhores serviços e preços mais baixos. Vários problemas relacionados podem ser formulados como um problema de otimização combinatória (COP). Uma vez que a maioria deles são *NP-Difíceis*, por causa de sua natureza combinatória, a descoberta de soluções ótimas por meio de métodos de solução exata muitas vezes é impraticável em um período de tempo razoável, especialmente para instâncias de grande porte que precisam de soluções de alta qualidade em tempos computacionais relativamente curtos. Em cenários realistas, onde os problemas são ainda mais complexos, restritos e dinâmicos, a necessidade crescente de tomada de decisão “instantânea” ou “rápida” refuta ainda mais o uso de abordagens exatas na vida real. Nessas circunstâncias, esta tese visa: (i) identificar COPs realistas de diferentes setores; (ii) desenvolver diferentes classes de abordagens de solução aproximada (heurísticas, heurísticas enviesadas-randomizadas, metaheurísticas, simheurísticas e otimização ágil) para resolver esses problemas de T&L inerentes de cidades inteligentes e sustentáveis; (iii) realização de uma série de experimentos computacionais para validar e medir o desempenho das abordagens desenvolvidas; e (iv) redigir conclusões quanto ao seu uso na solução dos problemas identificados. O novo conceito de “otimização ágil” (*agile optimization*) é introduzido, que se refere à combinação de heurísticas enviesadas-randomizadas com computação paralela para lidar com a tomada de decisões em tempo real. Além de solucionar esses problemas práticos, é apresentado um levantamento sobre os desafios na otimização de sistemas de compartilhamento de viagens –transporte compartilhado baseado em automóveis privados. Desafios e futuras linhas de pesquisa são abordados, especificamente no contexto de sistemas de telecomunicação e compartilhamento de caronas, onde a tomada de decisão imediata requer o emprego de estratégias de otimização ágil para fornecer soluções eficientes em tempo real.

Contents

Acknowledgements	v
Abstract	vii
Resumen	ix
Resum	xi
Resumo	xiii
1 Introduction	1
1.1 General Overview and Motivation	1
1.2 Objectives and Original Contribution	3
1.3 Summary of Research Outcomes	4
1.3.1 ISI-JCR Indexed Papers	5
1.3.2 Scopus Indexed Papers	5
1.4 Outline	6
2 Solving Methodologies	9
2.1 Heuristics	10
2.2 Metaheuristics	10
2.2.1 The Multi-Start	11
2.2.2 The Greedy Randomized Adaptive Search Procedure	12
2.2.3 The Iterated Local Search	14
2.2.4 The Variable Neighborhood Descent	15
2.3 Biased-Randomized Heuristics	16
2.4 Simheuristics	17
2.5 Agile Optimization	19
2.6 Conclusions	20
3 Applications in Transportation	23
3.1 The Omnichannel Vehicle Routing Problem	23
3.1.1 Literature Review	26
3.1.2 Problem Definition	28
3.1.2.1 Deterministic Omnichannel Vehicle Routing Problem	29
3.1.2.2 Stochastic Omnichannel Vehicle Routing Problem	30
3.1.3 Solution Method: From a Heuristic to a Simheuristic Approach	32

3.1.3.1	The Savings-Based Heuristic	32
3.1.3.2	Introducing a Local Search	34
3.1.3.3	Extending to a Biased-Randomized Algorithm	36
3.1.3.4	The Multi-Start Approach	38
3.1.3.5	The Simheuristic Approach	38
3.1.4	Computational Experiments and Results	40
3.1.4.1	Results for the Deterministic Omnichannel Vehicle Routing Problem	41
3.1.4.2	Results for the Stochastic Omnichannel Vehicle Routing Problem	44
3.1.5	Conclusions	47
3.2	The Vehicle Routing Problem with Optional Backhauls	49
3.2.1	Literature Review	51
3.2.2	Problem Definition	53
3.2.3	Solution Method: a Biased-Randomized Iterated Local Search	54
3.2.3.1	Generating an Initial Solution	55
3.2.3.2	Local Search Procedures	57
3.2.3.3	Perturbation Stage	57
3.2.3.4	Acceptance Criterion	58
3.2.4	Computational Experiments and Results	58
3.2.5	Conclusions	62
4	Applications in Humanitarian Logistics	67
4.1	The Two-Echelon Vehicle Routing Problem with Pick-up and Delivery	67
4.1.1	Literature Review	68
4.1.2	Problem Definition	69
4.1.3	Solution Method: an Agile Optimization Framework	71
4.1.4	Computational Experiments and Results	73
4.1.5	Conclusions	79
5	Applications in Retailing Industry	81
5.1	The Multi-Period Product Display Problem with Dynamic Attractiveness	81
5.1.1	Literature Review	84
5.1.1.1	The Product Assortment and Product Recommendation Problems	85
5.1.1.2	The Shelf Space Allocation Problem	87
5.1.2	Problem Definition	87
5.1.2.1	Mathematical Model	89
5.1.3	Solution Method: a GRASP and an ILS Approach	90
5.1.3.1	Extending the GRASP and ILS to a Biased-Randomized Algorithms	91
5.1.3.2	Constructing an Initial Solution for the Current Period	92
5.1.3.3	Improving the Solution of the Current Period	93

5.1.3.4	Perturbation Stage in the BR-ILS	93
5.1.3.5	Acceptance Criterion Stage in the BR-ILS	94
5.1.4	Computational Experiments and Results	95
5.1.4.1	Analysis of Results	96
5.1.4.2	Small-sized Instances and Limitations of Non-linear Solvers	96
5.1.4.3	Large-sized Instances with Fixed Profit Margins	97
5.1.4.4	Large-sized Instances with Varying Profit Margins (Selling Prices)	100
5.1.5	Conclusions	101
6	Applications in Production	103
6.1	The Hybrid Flow-Shop Vehicle Routing Problem	103
6.1.1	Literature Review	104
6.1.2	Problem Definition	106
6.1.3	Solution Method: From a Heuristic to a Metaheuristic	108
6.1.4	Solution Representation and Loading Strategy	109
6.1.5	Generating an Initial Solution	111
6.1.5.1	The BR-NEH Heuristic	111
6.1.5.2	The BR-SPTB Heuristic	111
6.1.5.3	The BR-bLPTB Heuristic	112
6.1.6	Neighborhood Structures	112
6.1.6.1	The SR_1 Neighborhood Structures	112
6.1.6.2	The SR_2 Neighborhood Structures	112
6.1.7	Computational Experiments and Results	114
6.1.7.1	Generation of Instances	114
6.1.7.2	Results of Small Instances	114
6.1.7.3	Results of Large Instances	116
6.1.8	Conclusions	119
7	Applications in Telecommunication	123
7.1	The Real-Time Facility Location Problem in Internet of Vehicle Scenarios . . .	123
7.1.1	Literature Review	126
7.1.2	Problem Definition	129
7.1.2.1	The Single-Period Internet of Vehicles Problem	129
7.1.2.2	The Multi-Period Internet of Vehicles Problem	131
7.1.3	Solution Method: From a Heuristic to a Agile Optimization Framework	136
7.1.3.1	Approaches for the Single-Period Internet of Vehicles Problem	137
7.1.3.2	Approaches for the Multi-Period Internet of Vehicles Problem	138
7.1.3.3	Agile Optimization	141
7.1.4	Computational Experiments and Results	142
7.1.4.1	Experimental Design	142
7.1.4.2	Results for the Single-Period Internet of Vehicles Problem . .	143
7.1.4.3	Results for the Multi-Period Internet of Vehicles Problem . .	148

7.1.5	Conclusions	151
8	Applications in Healthcare	155
8.1	The Application of Biased-randomized Algorithms for Supporting Logistics Decisions During the COVID-19 Crisis in 2020	155
8.1.1	The Barcelona's Makers Community	157
8.1.2	Routes Generation Process	159
8.1.3	Addressed Problems and Related Works	161
8.1.3.1	The Vehicle Routing Problem	161
8.1.3.2	The Team Orienteering Problem	163
8.1.3.3	Rich Pick-up and Delivery Routing Problems	164
8.1.4	Biased-Randomized Algorithms	165
8.1.5	Adapting Heavy Methods into Agile Algorithms	165
8.1.6	Solving Approach	166
8.1.7	Examples of Daily Routing Plans	167
8.1.8	Managerial Insights	171
8.1.9	Conclusions	172
9	Car-Sharing Systems	175
9.1	Introduction	175
9.2	Car Sharing Activities	177
9.2.1	Ride-sharing	177
9.2.2	Carpooling	178
9.3	Research Questions & Initial Classification	179
9.4	Exact Methods for Car Sharing Optimization	182
9.4.1	Ride-sharing	182
9.4.2	Carpooling	184
9.5	Metaheuristic Methods for Car Sharing Optimization	185
9.5.1	Ride-sharing	185
9.5.2	Carpooling	187
9.6	Simulation Methods for Car Sharing Management	188
9.6.1	Ridesharing	188
9.6.2	Carpooling	190
9.7	Performance Analysis of Ride-Sharing Systems	191
9.8	Challenges Related to Synchronization & Coordination	193
9.9	Challenges Related to Self-Driving and Electric Vehicles	196
9.10	Logistics Issues and Uncertainty Scenarios	197
9.11	Vehicle Technical Characteristics and Sustainability Issues	198
9.12	Hybrid <i>x-Heuristics</i> and Agile Algorithms for Ride-Sharing Problems	200
9.13	Conclusions	202

10 Conclusions, Future Research and Outcomes	205
10.1 Final Conclusions	205
10.2 Future Lines of Research	211
10.3 Research Outcomes	212
10.3.1 ISI-JCR Indexed Papers	212
10.3.1.1 Published	212
10.3.1.2 Under Peer-Review	213
10.3.2 Scopus Indexed Papers	213
10.3.2.1 Published	213
10.3.2.2 Under Peer-Review	214
Appendices	217
A.1 Complete Production	219
A.1.1 ISI-JCR Indexed Journal Papers	219
A.1.1.1 Published	219
A.1.1.2 Under Review	228
A.1.2 Scopus Indexed Journal Papers	232
A.1.2.1 Published	232
A.1.3 Peer-reviewed Conference Papers	236
A.1.3.1 Published	236
A.1.3.2 Under Review	239
B.2 Complementary Information	246
B.2.1 Mixed Integer Linear Programming Model of the Vehicle Routing Problem with Optional Backhauls	246
B.2.1.1 Mathematical Model	246
B.2.2 Mixed Integer Linear Programming Model of the Hybrid Flow-Shop Vehicle Routing Problem	248
B.2.2.1 Mathematical Model	249
B.2.2.2 A First Lower Bound for the HFS-VRP	251
Bibliography	255

List of Figures

1.1	Developed solution methodologies, considered application contexts, studied COPs and identified challenges during this thesis development.	4
2.1	Conflicting Criteria of Metaheuristics.	11
2.2	General Scheme of Simheuristics for Solving Stochastic COPs.	18
2.3	The general idea of the Agile Optimization framework: to run several executions of a biased-randomized heuristic in a parallel way.	20
2.4	Multi-dimensional comparison of different analytical approaches.	21
3.1	A general schema of the OCVRP.	25
3.2	A practical example of the OCVRP distribution system.	31
3.3	The flowchart of CWS_1	35
3.4	Possible merging cases in stage 2 (CWS_1).	36
3.5	The $2-opt$ movement.	37
3.6	The $2-opt$ movement for the OCVRP.	37
3.7	Gap between the best-found solutions (from MS_{BRLH}) and the MAC's results, for each inventory scenario.	44
3.8	Comparison of the solutions cost (OF value) from each solving approach, for each inventory scenario.	45
3.9	Comparison between MS_{BRLH} and MAC on solving large-sized instances.	46
3.10	Set of non-dominated solutions of problem instances $b17$ (a), $b37$ (b) and $b57$ (c).	48
3.11	Illustrating the vehicle routing problem with optional backhauls.	50
3.12	Gap between our OBSs and the BKSs for each penalization scenario (h_i).	61
3.13	Number of non-collected RTIs of our OBSs for each penalization scenario (h_i).	61
4.1	An example of a situation picture provided by the sensors.	70
4.2	An example of a 2-echelon distribution system.	72
4.3	Comparison between $BRLH$ and MAC on solving small-sized instances with a different number of parallel runs.	75
4.4	Comparison between AO and AH on solving large-sized instances with a different number of parallel runs.	79
4.5	Comparison between AO and MAC on solving large-sized instances with a different number of parallel runs.	80
5.1	A solution representation for a simple multi-period product display problem.	84
5.2	Comparison of percentage gaps w.r.t. the BKS achieved by each methodology.	99

5.3	Convergence chart for the BR-GRASP in a particular instance.	100
6.1	Combined production and distribution operations.	104
6.2	Combined production and distribution operations.	109
6.3	Flow-chart of the BR-VND algorithm.	110
6.4	Means plot and 95% Tukey confidence intervals for different combinations of test factors	119
7.1	The Internet of Vehicles Problem: connection of vehicles in motion to deployed RSUs in a city.	125
7.2	The MPIoVP modeled as an UFLP, where user and vehicles share the use of RSUs.	126
7.3	Heatmaps of traffic density at different time periods.	137
7.4	General Architecture of our approach.	141
7.5	Results obtained when varying the β value from 0 to 1.	143
7.6	The same instance is solved three times with different values of γ	145
7.7	Results obtained when varying the number of parallel runs.	147
7.8	Results obtained when varying the number of parallel runs, including the improvement rate of the solutions when comparing with De Armas et al. (2017)'s heuristic, given by the red cross intersection.	147
7.9	The variability of the average results for each scenario, in terms of gap.	150
7.10	The convergence of the OF value over the periods, in terms of gap, for problem instance 500.	153
8.1	The main goal of the Makers' community was to supply protective items to hospitals and healthcare centers.	156
8.2	Some of the pick-up locations geographically distributed in the area of Barcelona.	158
8.3	Overview of the material flow.	158
8.4	Daily node planning and interaction with research team.	159
8.5	Flowchart of the routes generation process.	160
8.6	An example of a VRP solution with a single central depot.	162
8.7	An example of a OVRP solution with different start and end depots.	163
8.8	An example of a TOP solution with different start and end depots.	164
8.9	Parallel execution of a biased-randomized algorithm.	167
8.10	The visual classification of the problems regarding their specifications.	170
8.11	Routes generated for the instance <i>apr-08</i> by VRP(a) and TOP(b) algorithms.	172
9.1	A visual representation of two possible ride-sharing rides.	178
9.2	A visual representation of a possible carpooling ride.	179
9.3	Number of citations per year indexed in WoS.	180
9.4	Number of scientific articles per year indexed in WoS.	181
9.5	The percentage of reviewed papers per solving methodology classification.	191
9.6	The percentage of reviewed papers regarding the addressed optimization objectives.	192

9.7 Main challenges and research opportunities related to ride-sharing operations. 194
9.8 Multi-dimensional comparison of different analytical approaches. 202

List of Tables

3.1	Comparison of the results obtained by the LH and MS_{BRLH} solution methods with those obtained by Abdulkader et al. (2018)'s methods (AH and MAC) in the tight inventory scenario.	42
3.2	Comparison of the results obtained by the LH and MS_{BRLH} solution methods with those obtained by Abdulkader et al. (2018)'s methods (AH and MAC) in the relaxed inventory scenario.	42
3.3	Comparison of the results obtained by the LH and MS_{BRLH} solution methods with those obtained by Abdulkader et al. (2018)'s methods (AH and MAC) in the abundant inventory scenario.	43
3.4	Parameter setup.	46
3.5	Analysis of the results obtained by the SIM_{BRLH} on scenarios of tight, relaxed and abundant inventory.	47
3.6	Comparison between our algorithm and previous works.	59
3.7	Found solutions for small-sized instances.	64
3.8	Found solutions for large-sized instances.	65
3.9	Comparison between VRPB and independent deliveries and pickups.	66
4.1	Comparison of results obtained by the different methodologies on solving small-sized instances.	74
4.2	Comparison of results obtained by the different methodologies in the scenario of tight inventory.	76
4.3	Comparison of results obtained by the different methodologies in the scenario of relaxed inventory.	77
4.4	Comparison of results obtained by the different methodologies in the scenario of abundant inventory.	78
5.1	Classification of main articles by problem and solving methodology.	88
5.2	Problem Parameters	96
5.3	Methodologies Parameters	96
5.4	Comparison of results between the non-linear solvers and our BR-GRASP approach (relaxed version of the problem)	97
5.5	Comparison of the results obtained by the proposed methodologies.	98
5.6	Comparison of the results obtained by the proposed methodologies.	102
6.1	Instance factor for the small and large instances.	114
6.2	Results of MILP and proposed LB for small instances.	116

6.3	Test factors for instances.	117
6.4	Average GAP over the proposed lower bound, grouped by instance characteristics.	118
6.5	CPU time (in seconds) of proposed algorithms for the large-sized instances.	120
7.1	Parameters setting for the numerical example of demands shifting.	136
7.2	Problem Parameters	143
7.3	Comparison of the results obtained by the proposed methodologies.	144
7.4	Computational results in terms of cost, facilities and completion time for the different scenarios.	149
8.1	Instances' inputs and outputs.	174
9.1	Classification of ride-sharing articles by version (static or dynamic) and solution methodology.	182

List of Algorithms

1	General Structure of the Multi-Start	12
2	General Structure of the GRASP	13
3	General Structure of the Iterated Local Search	14
4	General Structure of the Variable Neighborhood Descent	15
5	Savings-Based Heuristic	34
6	Biased-Randomized Selection	38
7	Biased-Randomized Savings-Based Heuristic	38
8	Multi-Start Biased-Randomized Savings-Based Heuristic	39
9	Simheuristic Approach	40
10	Biased-Randomized Iterated Local Search	55
11	Generating the Penalized Savings List	56
12	Agile Optimization Framework	73
13	Biased-Randomized Construction Stage	92
14	Refinement of the Configuration of Tables	93
15	Perturbation Stage	94
16	Credit-Based Acceptance Criterion	94
17	Neighborhood structures, LS_{C1} .	112
18	Neighborhood structures, LS_{P1} .	113
19	Neighborhood structures, LS_{CS1} .	113
20	Pseudo algorithm of the shifting model (main flow)	135
21	Pseudo implementation of the <i>shiftToward</i> function.	135
22	Extended Heuristic for solving the IoVP	139
23	Extended Heuristic for solving the MPIoVP	140
24	Example of one heuristic-based approach employed.	168

Nomenclature

2E-VRP	Two-Echelon Vehicle Routing Problem
ACO	Ant Colony Optimization
AFV	Alternative Fuel Vehicle
ALNS	Adaptive Large Neighborhood Search
AO	Agile Optimization
AV	Autonomous Vehicle
BH	Backhaul
BKS	Best-Known Solution
bLPTB	Backward Largest Processing Time Bottleneck
BR	Biased-Randomized
BR-GRASP	Biased-Randomized Greedy Randomized Adaptive Search Procedure
BR-ILS	Biased-Randomized Iterated Local Search
BR-VND	Biased-Randomized Variable Neighborhood Descent
BRH	Biased-Randomized Heuristic
BRKGA	Biased Random-Key Genetic Algorithm
bSPTB	Backward Shortest Processing Time Bottleneck
CF	Collaborative Filtering
CLSM	Complete Local Search with Memory
CO ₂ e	Carbon Dioxide-Equivalent Emissions
COP	Combinatorial Optimization Problem
CVRP	Capacitated Vehicle Routing Problem
CWS	Clarke & Write Savings
ESCI	Emerging Sources Citation Index
EV	Electric Vehicle
FAL	First-Aid Location
FLP	Facility Location Problem
GA	Genetic Algorithm
GRASP	Greedy Randomized Adaptive Search Procedure
HFS	Hybrid Flow-Shop

HFS-VRP Hybrid Flow-Shop Vehicle Routing Problem
ICEV Internal Combustion Engine Vehicle
ICT Information and Communication Technologies
ILP Integer Linear Programming
ILS Iterated Local Search
IMCNFP Integer Multiple Commodity Network Flow Problem
IoV Internet of Vehicles
IoVP Internet of Vehicles Problem
IP Integer Programming
IRP Inventory Routing Problem
IS Insertion Heuristic
ISI Institute for Scientific Information
JCR Journal Citation Reports
LB Lower Bound
LH Linehaul
LNS Large Neighborhood Search
LPT Longest Processing Time
LPTB Largest Processing Time Bottleneck
LS Local Search
MAC Multi-Ant Colony Algorithm
MCS Monte Carlo Simulation
MILP Mixed-Integer Linear Programming
MIP Mixed-Integer Programming
MPIoVP Multi-Period Internet of Vehicles Problem
MPPDPDA Multi-Period Product Display Problem with Dynamic Attractiveness
MS Multi-Start
MTL Maximum Tour Length
NFV Network Functions Virtualization
NVD Nearest Vehicle Dispatch
OBS Our Best Solution
OCVRP Omnichannel Vehicle Routing Problem
P2P Peer-to-Peer
PDP Pickup and Delivery Problem
PL Pharmaceutical Laboratory
PSO Particle Swarm Optimization

QoS	Quality of Service
RCL	Restricted Candidates List
RDPD	Rich Pick-up and Delivery Problem
RSU	Roadside Unit
RTI	Returnable Transport Item
SA	Simulated Annealing
SAV	Shared Autonomous Vehicle
SBS	Small Base Station
SCI	Science Citation Index
SL	Savings List
SOCVRP	Stochastic Omnichannel Vehicle Routing Problem
SPA	Sequential Pattern Analysis
SPLP	Simple Plant Location Problem
SPT	Shortest Processing Time
SPTB	Shortest Processing Time Bottleneck
SSCI	Social Science Citation Index
T&L	Transportation and Logistics
TOP	Team Orienteering Problem
TS	Tabu Search
TSP	Traveling Salesman Problem
UB	Upper Bound
UCL	Unrestricted Candidates List
UFLP	Uncapacitated Facility Location Problem
V2V	Vehicle-to-Vehicle
V2X	Vehicle-to-Everything
VND	Variable Neighborhood Descent
VNS	Variable Neighborhood Search
VRP	Vehicle Routing Problem
VRPB	Vehicle Routing Problem with Backhauls
VRPBTW	Vehicle Routing Problem with Backhauls with Time Windows
VRPDSP	Vehicle Routing Problem with Deliveries and Selective Pick-ups
VRPOB	Vehicle Routing Problem with Optional Backhauls
VRPSPD	Vehicle Routing Problem with Simultaneous Pick-up and Delivery
WoS	Web of Science

Chapter 1

Introduction

1.1 General Overview and Motivation

Transportation and logistics (T&L) activities play a vital role in the development of many businesses from different industries. With the increasing number of people living in urban, metropolitan, and peri-urban areas, several urban problems have been potentialized, such as traffic congestion, pollution, risk of accidents, impaired vehicular and pedestrian flow. Thanks to globalization and the expansion of both the on-demand economy (services) and the e-commerce activity (products), for example, this population is constantly requiring a large number of services from transportation, delivery of products, medical and educational services. Customers, on the other hand, are increasingly familiar with the use of new technologies, demanding better services and lower costs, and more environmentally conscious. Urban logistics, therefore, became a critical decision in which many companies and logistics service providers from different industries are willing to improve the efficiency of their operations.

The pressure to improve these logistics operations is derived from many circumstances, for instance: *i*) the global market is becoming increasingly competitive; *ii*) the customers are more informed, demand higher quality in products and services, and lower costs; *iii*) companies are adopting new types of operations, such as just-in-time, lean operations, time compression, flexible manufacturing, mass customization, virtual operations; *iv*) today, there are significant improvements in communication technologies, which allow new practices and models of commerce and distribution, such as e-commerce, shared knowledge systems; *v*) managers recognize both the strategic and tactical importance of the supply chain; *vi*) decisions in transportation are changing, mainly due to the increased traffic and congestion on roads, concerns about environmental issues, such as air quality and pollution; and so on (Waters, 2003). In this scenario, it is clear that T&L activities play a major role in the development of the so-called smart sustainable cities (Bibri and Krogstie, 2019).

On the one hand, smart cities combine the use of information and communication technologies (ICT) to sustainability to solve urban problems towards inducing economic development and improve the life quality of the population (Albino et al., 2015). On the other hand, sustainability refers to the ability of a city to maintain the balance of the ecosystem, while attending and performing city operations (Silva et al., 2018). This ecosystem, in turn,

comprises different attributes, which range from infrastructure and governance to pollution, waste, energy, climate change, and economics, for example.

Environmentally speaking, T&L activities are one of the major contributors to the greenhouse effect globally. In 2018, 28% of total carbon dioxide-equivalent emissions (CO₂e) in the United States have been entailed to transportation, being light-duty vehicles responsible for 59% of them (United States Environmental Protection Agency, 2020). In Europe, on the other hand, 27% of total greenhouse gas emissions originated from the transport sector, being 2.2% higher when compared with the previous year. In 2018, transportation was responsible for almost 30% of CO₂e, of which 72% comes from road transportation (European Environment Agency, 2019). With the purpose of mitigating related problems, such as greenhouse effects and global warming, the European Union has developed strategic plans for low-emission mobility in which the primary pillar is based on increasing the efficiency of the transport system by benefiting from digital technologies, smart pricing, and further encouraging the shift to lower emission sustainable transportation modes (European Commission, 2016).

Apart from playing a major role in the environment, T&L activities compose a considerable part of products' total cost in the supply chain. Supply chain refers to a set of processes that transform raw materials into final products, from the production to their distribution to final users (Mentzer et al., 2001). According to Lapinskaitė and Kuckailytė, 2014, about 55% of a product's total cost is derived from supply chain activities, being transportation a considerable part of it. In healthcare, logistics cost represents 38% of the total expenses, while 5% and 2% in retail and electronic industries, respectively (Johnson, 2015). One of the main reasons that contribute to this gap refers to the inefficient utilization of cargo vehicles and warehouses, besides the employment of a unique distribution network in distribution systems. In healthcare and relief operations, for instance, the distribution of medical supplies is a crucial activity that opens the door to challenging studies in the areas of VRPs, inventory optimization, and simulation (Landry and Beaulieu, 2013). In this context, improving the efficiency of logistics activities in these industries can increase not only the quality of care and reduce costs for providers (Moons et al., 2019), but also provide end-users (customers) with better services and lower costs. Therefore, the need for increasing the effectiveness and sustainability of T&L activities is clear, whose development has been possible due to recent advances in ICT.

Many of these problems, from tactical, strategic, and operational levels in the supply chain, can be formulated as a combinatorial optimization problem (COP). This class of decision-making problems comprises numerous industrial processes and planning activities from diverse application contexts, such as vehicle routing, facility location, and scheduling problems. Most of the important COPs are *NP-Hard* (Colomi et al., 1996), which means that they cannot be solved exactly in polynomial time (Glover and Kochenberger, 2006) since the space of potential solutions to these problems grows exponentially as the instance size increases. In this way, because of the combinatorial nature of these problems, the finding of optimal solutions through exact solution methods is often impractical in a reasonable

amount of time, especially for large-size instances that need high-quality solutions in relatively short computational times. In realistic settings, where problems are even more complex, constrained, and dynamic, the increasing need for ‘instant’ or ‘fast’ decision-making further refutes the use of exact approaches in real life. Under these circumstances, heuristic and metaheuristic approaches have been gaining tremendous popularity due to their satisfactory response when tackling this class of problems, being them excellent alternatives to exact methods (Talbi, 2009). A biased-random behavior can be incorporated in these methodologies, which transforms a deterministic algorithm into a probabilistic one without losing the logic behind the original algorithm (Juan et al., 2013a). However, there still situations in which solutions must be generated in real-time, e.g., in the event of disasters, where real-time optimization can be a life-saving differential. To cope with these real-time decision-making systems, this thesis introduces the concept of ‘agile optimization,’ which refers to the combination of biased-randomized heuristics with parallel computing to deal with real-time decision-making. Moreover, real-life based problems are often fraught with uncertainties, such as processing times, demands, travel times, breakdown of vehicles, and traffic control. To this aim, the simheuristics, which are powerful approaches designed for coping with optimization problems under uncertainty by combining simulation techniques with heuristics and metaheuristics methodologies (Juan et al., 2015a), are taken into account.

Due to the heightened need for solving these real-life based problems, from social to economical perspectives, this thesis aims at developing the five mentioned classes of solution approaches (heuristics, biased-randomized heuristics, metaheuristics, simheuristics, and agile optimization) to solve different T&L problems, inherits in smart and sustainable cities. Apart from solving these practical problems, a survey on the challenges in optimization for automobile-based shared transportation systems –ride-sharing and carpooling– is presented. Figure 1.1 presents the high-level structure of this thesis. The following information is provided: the developed solution methods, the considered application contexts, the studied COPs, and, finally, the main identified challenges.

1.2 Objectives and Original Contribution

This thesis focuses on studying the applications of biased-randomized algorithms –including heuristic and metaheuristic approaches, simheuristics, and agile optimization techniques– to solve T&L problems from diverse application areas. Based on this main objective, the following original contributions and research results are attained:

- Realistic business and industrial optimization problems from the integrated logistics, especially those belonging to the transportation of goods, are formally addressed and defined.
- Efficient solution algorithms, based on biased-randomization of heuristics, metaheuristics, and simheuristics, are proposed and developed for solving the identified optimization problems.

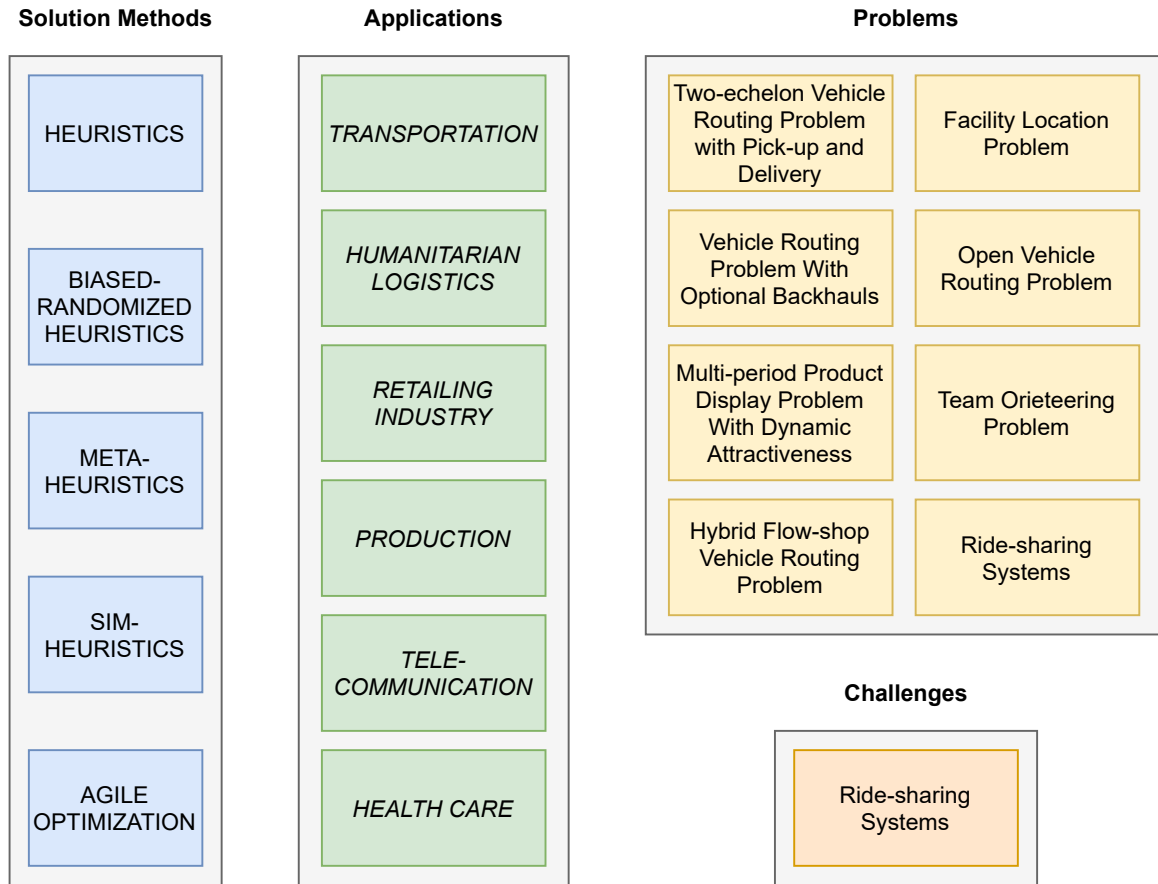


Figure 1.1: Developed solution methodologies, considered application contexts, studied COPs and identified challenges during this thesis development.

- An original approach –agile optimization– is proposed to cope with real-time decision-making, such as those required in relief operations or telecommunication systems.
- A series of computational and numerical experiments are conducted to evaluate the performance of the developed algorithms when applied to realistic and large-sized problem instances.
- Managerial insights and conclusions about the potential advantages of using the developed algorithms in complex and real-life decision-making processes are drawn.
- Finally, an in-depth study in the context of new modes of passenger transport, such as carpooling and ride-sharing systems, is conducted. As a result, new lines of research are presented.

1.3 Summary of Research Outcomes

The core of this thesis is based on the several research outcomes published in different ISI-JCR/Scopus indexed journals and proceedings of international peer-reviewed conferences. The relevant publications are mentioned at the beginning of their corresponding chapters, being their contents the pillar of the application context under study. In the following, the

research articles which are currently published are enumerated. It includes four research papers published in ISI-JCR indexed journals (Section 1.3.1), and four Scopus indexed journals/conference proceedings (Section 1.3.2). The complete scientific production, including those under peer-reviewing process (u.r.), is presented in Chapter 10. Moreover, the cover page of each article is presented in Appendix A.1.

1.3.1 ISI-JCR Indexed Papers

1. **Martins, L. C.**; Hirsch, P.; Juan, A. A. (2020): [Agile optimization of a two-echelon vehicle routing problem with pickup and delivery](#). *International Transactions in Operational Research*, 28(1), 201-221.
2. Londoño, J. C.; Tordecilla, R.; **Martins, L. C.**; Juan, A. A. (2020): [A biased-randomized iterated local search for the vehicle routing problem with optional backhauls](#). *TOP*, 1-30.
3. Bayliss, C.; **Martins, L. C.**; Juan, A. A. (2020): [A two-phase local search with a discrete-event heuristic for the omnichannel vehicle routing problem](#). *Computers & Industrial Engineering*, 148, 106695.
4. **Martins, L. C.**; de la Torre, R.; Corlu, C. G.; Juan, A. A.; Masmoudi, M. A. (2021): [Optimizing ride-sharing operations in smart sustainable cities: Challenges and the need for agile algorithms](#). *Computers & Industrial Engineering*, 153, 107080.
5. **Martins, L. C.**; Gonzalez, E.; Hatami, S.; Juan, A. A.; Montoya, J. (2021): [Combining Production and Distribution in Supply Chains: the Hybrid Flow-Shop Vehicle Routing Problem](#). *Computers & Industrial Engineering*, 159, 107486.
6. **Martins, L. C.**; Tarchi, D.; Juan, A.; Fusco, A. (2021): [Agile Optimization for Real-Time Facility Location Problem in Internet of Vehicle Scenarios](#). *Networks*.
7. **Martins, L. C.**; Tordecilla, R.; Castaneda, J.; Juan, A.; Faulin, J. (2021): [Electric Vehicle Routing, Arc Routing, and Team Orienteering Problems in Sustainable Transportation](#). *Energies*, 14(16), 5131.
8. Tordecilla, R.; **Martins, L. C.**; Panadero, J.; Copado-Méndez, P. J.; Perez, E.; Juan, A. A. (2021). [Fuzzy Simheuristics for Optimization Problems in Transportation: dealing with stochastic and fuzzy uncertainty](#). *Applied Sciences*, 11(17), 7950.

1.3.2 Scopus Indexed Papers

1. Marmol, M.; **Martins, L. C.**; Hatami, S.; Juan, A. A.; Fernandez, V. (2020): [Using biased-randomized algorithms for the multi-period product display problem with dynamic attractiveness](#). *Algorithms*, 13(2), 34.
2. **Martins, L. C.**; Bayliss, C.; Juan, A. A.; Panadero, J.; Marmol, M. (2020): [A savings-based heuristic for solving the omnichannel vehicle routing problem with pick-up and delivery](#). *Transportation Research Procedia*, 47, 83-90.

3. **Martins, L. C.**; Bayliss, C.; Copado-Méndez, P. J.; Panadero, J.; Juan, A. A. (2020): [A Simheuristic Algorithm for Solving the Stochastic Omnichannel Vehicle Routing Problem with Pick-up and Delivery](#). *Algorithms*, 13(9), 237.
4. Bayliss, C.; Copado-Méndez, P. J.; Panadero, J.; Juan, A. A.; **Martins, L. C.** (2020): [A Simheuristic-Learnheuristic Algorithm for the Stochastic Team Orienteering Problem with Dynamic Rewards](#). *Proceedings – 2020 Winter Simulation Conference (WSC)*, 1254-1264.
5. Tordecilla, R.; **Martins, L. C.**; Saiz, M.; Copado-Méndez, P. J.; Panadero, J.; Juan, A. A. (2021): [Agile Computational Intelligence for supporting Hospital Logistics during the COVID-19 Crisis](#). *Computational Management*, 383-407.

1.4 Outline

This thesis is structured as follows:

- Chapter 2 addresses the different classes of solution methodologies that have been proposed and developed to solve the identified integrated logistics problems. These methodologies belong to the class of approximate optimization methods, which contrast to exact approaches. Exact methods can find the optimal solutions but are often limited by the complexity of real-life problems and their magnitude. Despite not guaranteeing the finding of optimal solutions, the approximate solving approaches are able to generate high-quality solutions for a vast of COPs, apart from being capable to deal with many determinants, such as stochasticity, dynamism, scalability, among others.
- Chapters 3 to 8 presents the several application contexts in which the aforementioned solution methods have been proposed to cope with. The applications are in: (i) transportation, (ii) humanitarian logistics, (iii) retailing industry, (iv) production, (v) telecommunication, and (vi) health care. For each application context, the corresponding optimization problems are introduced, followed by their formal description, relevant literature review, solution methodologies, results and conclusions.
- Chapter 9 introduces an in-depth study concerning the challenges of optimizing ride-sharing operations in smart sustainable cities. These new modes of transportation have become a trend in recent years mainly due to advances and ICT and the competitive ground passenger transportation market. A series of related challenging operations are addressed and discussed, such as the synchronization issues and the use of electric and autonomous vehicles in these systems.
- Chapter 10 provides a series of conclusions concerning the research done and points out future challenging research based on the addressed solution methods and applications.

-
- Appendix [A.1](#) presents the cover page of the research outcomes which are published and under peer-reviewing process to date, categorized by the type of production, i.e., conference article or journal article.
 - Appendix [B.2](#) introduces additional information regarding the mathematical formulation and programming models of the identified COPs, whenever their reading is relevant to support the analysis of results.

Chapter 2

Solving Methodologies

Many problems from different theoretical and practical perspectives aim at finding the best (or optimal) configuration for a given set of parameters in order to achieve a specific goal (Papadimitriou and Steiglitz, 1998). This process of finding optimal solutions in a well-defined discrete problem space refers to combinatorial optimization. Accordingly, combinatorial optimization problems (COPs) can be seen as decision-making problems where a finite number of alternative feasible solutions exist (Hoffman and Ralphs, 2013), being them commonly found in both industrial processes and planning activities from diverse application contexts. Because this class of problems is composed of a discrete space of feasible solutions, the trivial way to solve a COP is to enumerate all the possible feasible solutions from the solution space and select the best one, according to its objective. However, in many cases, the number of solutions to be enumerated is intractable due to the combinatorial nature of this type of problem.

Depending on its complexity, a COP can be solved by an exact, or approximate, or even by a hybrid methodology. Exact methodologies are able to obtain high-quality solutions and guarantee their optimality. However, when dealing with an *NP-hard* problem, such those decision-making problems in which more than one conflicting objective and constraint are involved, their use is often inappropriate to solve medium- and large-size problem instances, since the search is carried out over the whole interesting search space that grows exponentially according to the instance magnitude (Talbi, 2009).

As introduced in Section 1, most of the real-life T&L problems are known as *NP-hard*. Under this circumstance, the use of approximate approaches, which are, apart from being capable to deal with large- and medium-sized problems, able to provide high-quality solutions within a reasonable short computational time, has emerged substantially over the last decades. Within this class of approaches for solving COPs, we can mention the use of heuristics and metaheuristics algorithms, which have become the default standard when dealing with rich and realistic problems. A decade ago, skewed probability distributions were incorporated into heuristic algorithms to bias its behavior (Juan et al., 2010). Since then, biased-randomized algorithms have been employed for solving different COPs in diverse areas, providing solutions with an interesting balance in respect to the solution quality and computational time. More recently, simheuristics approaches, which combine metaheuristics with simulation techniques (Juan et al., 2015a), have been proved to be an efficient

technique to cope with problems under uncertainty scenarios. In this thesis, apart from addressing solution methods from heuristics to simheuristics to solve T&L COPs, the concept of ‘agile optimization’ is introduced. Agile optimization methods extend the benefits of BR heuristics by embedding them into a parallel framework, where multiple instances of a BR heuristic is run concurrently, then reducing the overall processing time of the methodology. Consequently, the resulting approach is able to deal with large-sized and dynamic problems with higher utilization of processing resources. The next sections describe the class of solution methods employed and developed in this thesis.

2.1 Heuristics

Heuristics can be described as a sequence of logical steps to achieve a certain target and solve a specific problem. Usually, they consist of simple and deterministic moves or actions, which implies that the same final output is obtained whenever this process starts from the same initial point, with the same parameter settings, i.e., the same solve trajectory is executed. Although the heuristics are able to generate good and feasible solutions in a very short computational time to a majority of COPs, these simple and fast solving methodologies do not ensure an ideal balance between diversification and intensification of the solution space. While the term ‘diversification’ refers to the exploration of the search space, the term ‘intensification’ refers to the exploitation of the accumulated search experience. For this reason, heuristics are, for instance, frequently employed to generate initial solutions for a vast of more robust methodologies, such as metaheuristics and simheuristics.

2.2 Metaheuristics

Different from heuristics, which make use of experience-based techniques to solve a problem, metaheuristics are problem-independent algorithmic frameworks that operate at a higher level of abstraction. These frameworks provide a set of guidelines to develop heuristic optimization algorithms (Sörensen and Glover, 2013). In this way, once a metaheuristic is developed, different problem domains can be tackled by replacing the set of low-level heuristic optimization algorithms and the evaluation function that indicates the quality of a given solution (Talbi, 2009).

As introduced, heuristics algorithms are not able to guarantee an ideal balance between diversification and intensification of the solution space. Metaheuristics, on the other hand, are capable to achieve an appropriate and dynamic balance between the intensification of the search experience gathered so far and the diversification of unvisited or relatively unexplored search space regions. According to Blum and Roli (2003), this balance is important to quickly identify promising regions in the search space and, consequently, avoid wasting too much time in regions already explored or which do not provide high-quality solutions.

Metaheuristics algorithms can be categorized in terms of several properties. According to Talbi (2009), their classification mainly relies on:

- *Nature inspired versus non-nature inspired;*

- *Memory usage versus memoryless methods;*
- *Deterministic versus stochastic;*
- *Population-based search versus single-solution based search;*
- *Iterative versus greedy.*

Accordingly, the trade-off between diversification and intensification of metaheuristics, regarding their classification, can be seen in Figure 2.1. In general, single-solution based metaheuristics are more exploitation-oriented, whereas basic population-based metaheuristics are more exploration-oriented. However, the most appropriate metaheuristic to efficiently solve a problem should take into account not only its own characteristics, but also its ability to adapt to a particular environment, avoid getting stuck at local optimum and exploit the basic structure of the problem (Feo and Resende, 1995).

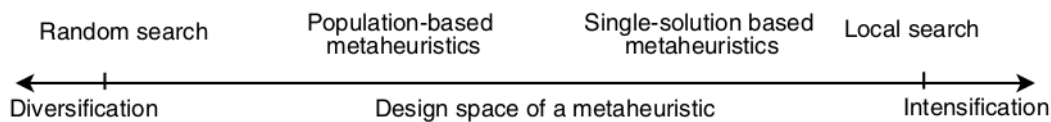


Figure 2.1: Conflicting Criteria of Metaheuristics.
Source: Talbi (2009).

Based on these properties, the application of metaheuristic algorithms to solve COPs has increased a lot in the last years, as well as its field of research. These algorithms have been proved to be very satisfactory in many areas, such as transportation (e.g., Lourenço et al. (2001), Gendreau et al. (2002), and Song et al. (2020), telecommunication (e.g., Martins and Ribeiro (2006) and Fernandez et al. (2018)), healthcare (e.g., Hiermann et al. (2015) and Fathollahi-Fard et al. (2020)), medicine and biology (e.g., Nakib and Talbi (2017), Diniz et al. (2019), and Ushizima et al. (2020)), finance (e.g., Soler-Dominguez et al. (2017) and Ansari et al. (2020)), and so on. Among the well-known metaheuristics, this thesis highlights the use of Multi-Start (MS) (Martí et al., 2016), Greedy Randomized Adaptive Search Procedure (GRASP) (Feo and Resende, 1995; Resende and Ribeiro, 2003), Iterated Local Search (ILS) (Lourenço et al., 2003), and Variable Neighborhood Search (VNS) (Hansen and Mladenović, 2003) approaches for solving the addressed applications and related COPs.

2.2.1 The Multi-Start

Multi-start methods can be considered as the core of many metaheuristics frameworks. These iterative methods are composed of two main stages: (i) the solution construction; and (ii) the solution improvement, being the later not necessarily mandatory. Each global iteration produces a solution, which is usually a local optima from a region of the solution space, and the best overall solution is returned by the algorithm (Martí et al., 2016). Although the employment of local search methods for improving a new solution is not mandatory, the

consideration of the second stage in MS approaches yields high-quality solutions, by creating a balance between search diversification and search intensification. However, depending on the problem, constructing new solutions is more effective than applying local search procedures. For instance, in some constrained problems, the finding of feasible solutions through neighborhood movements is harder than constructing new feasible solutions from scratch (Martí et al., 2013). Therefore, MS methods are structurally simple approaches that can provide a suitable framework to develop methodologies for solving both constrained and non-constrained problems.

In Pseudocode 1, the overall structure of an MS algorithm is presented for a general minimization problem. As described, its main idea is based on creating a new solution (line 3), applying a local search mechanism to improve its quality (line 4), and updating the best-found solution (lines 5-7). These processes are repeated until a stop criterion is met (line 2), and the best overall solution is returned by the method (line 10).

Pseudocode 1: General Structure of the Multi-Start

```

1 Function MS():
2    $f^* \leftarrow \infty$ 
3   while stop condition is not satisfied do
4      $current_{sol} \leftarrow \text{constructFeasibleSolution}()$ 
5      $current'_{sol} \leftarrow \text{improveSolution}(current_{sol})$ 
6     if  $f(current'_{sol}) < f^*$  then
7        $best_{sol} \leftarrow current'_{sol}$ 
8        $f^* \leftarrow f(current'_{sol})$ 
9     end
10  end
11  return  $best_{sol}$ 
12 End

```

2.2.2 The Greedy Randomized Adaptive Search Procedure

The Greedy Randomized Adaptive Search Procedure consists of another metaheuristic that is based on an MS or iterative process, in which each iteration is composed by the construction of a solution and its refinement through a local search method. However, its core difference from an MS approach relies on the construction stage of a feasible solution, which employs a restricted candidates list (RCL) to select the candidate elements to be incorporated into the partial solution (Resende and Ribeiro, 2003). In other words, the set of candidate elements, which is initially made up of all elements that can be incorporated into the partial solution under construction without destroying feasibility, is restricted by creating a restricted candidate list that is composed of the best elements, i.e., those which yield to the smallest increment in the solution cost. The size of the RCL is strictly defined by: (i) a cardinal parameter p , which defines the size of the list; or (ii) a threshold parameter $\alpha \in [0, 1]$, which implies that only elements that offer a minimum quality, established by α , can be inserted. In other words, only elements $e \in E$ with $c(e) \in [c_{min}, c_{min} + \alpha(c_{max} - c_{min})]$

are inserted into the RCL, where c_{min} and c_{max} are the current minimum and maximum incremental cost, respectively. When $\alpha = 0$, the greedy approach is invoked, while $\alpha = 1$ is equivalent to the random selection. Within this RCL, which is reduced, the elements are randomly selected to be part of the solution, therefore, guaranteeing the ‘greedy randomized’ purpose of the strategy. The elements to be inserted into the RCL are evaluated by a greedy evaluation function that usually represents the incremental increase in the cost function if these elements are inserted into the solution. Whenever the selected element is incorporated into the partial solution under construction, the list of candidate elements is updated and re-evaluated, which refers to the ‘adaptive’ feature of the methodology. Similarly to the MS approach, this iterative process is also controlled by a maximum number of iterations.

Pseudocode 2 describes the structure of a GRASP algorithm. In this case, compared to the MS approach, the construction of feasible solutions at each iteration (line 4) is extended by considering the creation of the RCL. This process is presented between lines 12 and 18, where candidate elements are evaluated (line 14), the RCL is created according to the parameter α (line 16), and the empty solution is constructed by incorporating elements randomly chosen from the RCL (lines 16 and 17). Once introduced into the solution, the element is removed from the candidates’ list, and the remaining elements are re-evaluated according to their current incremental cost. This process is repeated until the solution is formed.

Pseudocode 2: General Structure of the GRASP

Data: set of candidate elements E , threshold parameter α

```

1 Function GRASP( $\alpha, E$ ):
2    $f^* \leftarrow \infty$ 
3   while stop condition is not satisfied do
4      $current_{sol} \leftarrow \text{constructGreedyRandomizedFeasibleSolution}(\alpha)$ 
5      $current'_{sol} \leftarrow \text{improveSolution}(current_{sol})$ 
6      $\text{updateBestSolution}(current'_{sol}, best_{sol})$ 
7   end
8   return  $best_{sol}$ 
9 End
10 Function  $\text{constructGreedyRandomizedFeasibleSolution}(\alpha, E)$ :
11    $current_{sol} \leftarrow \emptyset$ 
12   Evaluate the incremental cost of candidate elements  $E$ 
13   while  $current_{sol}$  is not complete do
14      $RCL \leftarrow \text{constructRCL}(\alpha, E)$ 
15     Randomly select  $pos \in [1, \dots, |RCL|]$ 
16      $current_{sol} \leftarrow current_{sol} \cup \{RCL[pos]\}$ 
17      $E \leftarrow E \setminus \{RCL[pos]\}$ 
18     Re-evaluate the incremental cost of candidate elements
19   end
20   return  $current_{sol}$ 
21 End

```

2.2.3 The Iterated Local Search

So far, both MS and GRASP approaches rely on iteratively constructing solutions from the ground up, and refining these solutions in order to enhance their quality. Although belonging to the class of metaheuristics which are based on construction and improvement of solutions iteratively over time, the Iterated Local Search is mainly characterized by the introduction of a perturbation stage in this process to avoid solutions being trapped in local optima (Lourenço et al., 2003). Ideally, the perturbation is designed to perform movements that local search mechanisms must avoid. In other words, the perturbation is aimed to perform neighborhood movements reasonably large enough to explore different regions of the solution space. Local search methods, on the other hand, are more focused on performing small movements to exploit (i.e., intensify) the current space where the solution lies. Therefore, the efficiency of an ILS procedure strongly depends on the size of the perturbation neighborhood: while small movements preclude the exploration of new solutions, large movements culminate randomly starting points. Therefore, both the solution quality and computational effort strongly depends on the perturbation procedure. One way to avoid exploring already found local optimum is to use historical information. Apart from this stage, the acceptance criterion of ILS is another important step to provide a balance between intensification and diversification of the solution space. In the literature, the acceptance criterion of the ILS is frequently modeled as that one belonging to the simulated annealing (SA) metaheuristic, which employs an acceptance probability for accepting candidate solutions (Henderson et al., 2003).

In Pseudocode 3, the structure of an ILS method is presented. Differently to MS and GRASP, a single solution is constructed (line 3) and improved through a local search procedure (line 4) before starting its main life cycle. Successive perturbation-improvement procedures are applied in this refined solution (lines 6 and 7), which are possibly updated according to the defined acceptance criterion (lines 8-10). The perturbation, improvement, and acceptance of solutions are executed until a stop criterion is achieved.

Pseudocode 3: General Structure of the Iterated Local Search

```

1 Function ILS():
2    $f^* \leftarrow \infty$ 
3    $initial_{sol} \leftarrow \text{constructFeasibleSolution}()$ 
4    $best_{sol} \leftarrow \text{improveSolution}(initial_{sol})$ 
5   while stop condition is not satisfied do
6      $current_{sol} \leftarrow \text{perturbation}(best_{sol}, \text{history})$ 
7      $current'_{sol} \leftarrow \text{improveSolution}(current_{sol})$ 
8      $\text{acceptanceCriterion}(current'_{sol}, best_{sol}, \text{history})$ 
9   end
10  return  $best_{sol}$ 
11 End

```

2.2.4 The Variable Neighborhood Descent

Neighborhood structures play a crucial role in the performance of a local search method in any of the introduced metaheuristic frameworks. Similar to the ILS, which makes use of a perturbation procedure to escape from local optima, other metaheuristics incorporate different strategies to explore different regions of the solution space. For this purpose, the Variable Neighborhood Search metaheuristic operates by performing systematic changes of neighborhood structures within a local search mechanism, contrasting with most local search heuristics that use only one neighborhood structure to explore the space of possible solutions (Hansen and Mladenović, 2003). The VNS, therefore, rather than follow a trajectory, consists of exploring increasingly distant/large neighborhoods of the incumbent solution and jumping from this solution to a new one whenever an improvement has been achieved (Hansen and Mladenović, 2001). Its basic idea is based on the repetitive application of a local search routine to get from these neighboring solutions to different local optima, given that a local minimum for one neighborhood structure is not necessary a local minimum with respect to another one. The case in which the change of neighborhoods is performed in a deterministic manner results in the Variable Neighborhood Descent (VND), which means that the diversification part of VNS –the exploration of the search space– is replaced by the intensification phase –the exploitation of the accumulated search experience.

Pseudocode 4 shows the structure of the classical VND. In this case, a set of u_{max} neighborhood structures guides the search for efficient solutions. Different local search movements are systematically applied to an initial solution, starting from the first neighborhood structure (line 2), generating a new solution sol' that belongs to the neighborhood under consideration (line 4). Whenever a better solution is found (line 5), the former initial solution is updated (line 6), and the search is re-started from the first movement structure (line 7). If the current neighborhood cannot find a better solution, the search continues with the same solution but applying the posterior neighborhood structure (line 9). This refinement process is concluded when all the movements are applied to the current solution, but no improvements are found (line 3).

Pseudocode 4: General Structure of the Variable Neighborhood Descent

Data: initial solution sol , number of neighborhood structures u_{max}

```

1 Function VND( $sol, u_{max}$ ):
2    $u \leftarrow 1$ 
3   while  $u \leq u_{max}$  do
4      $sol' \leftarrow$  the best-found solution in neighborhood  $N_u(sol)$ 
5     if  $f(sol') < f(sol)$  then
6        $sol \leftarrow sol'$ 
7        $u \leftarrow 1$ 
8     else
9        $u \leftarrow u + 1$ 
10    end
11  end
12  return  $sol$ 
13 End

```

2.3 Biased-Randomized Heuristics

As introduced in Section 2.1, heuristic algorithms are often designed under deterministic premises. One way to overcome this deterministic particularity regards the inclusion of a random behavior to guide the heuristics' search process. Accordingly, related strategies employ the uniform distribution to incorporate the desired random search. However, despite being able to explore different areas from the solution space, this 'blind' search often destroys the logical idea behind the heuristics.

Biased-randomized (BR) algorithms belong to a class of optimization techniques that transform a deterministic algorithm into a probabilistic one without losing the logic behind the heuristic. They incorporate non-symmetric random sampling –through skewed probability distributions– to introduce a non-uniform random to diversify the behavior of a base constructive heuristic (Estrada-Moreno et al., 2019). These skewed probability distributions allow the assignment of probabilities to actions (or elements) to be chosen and performed according to a certain criterion (Juan et al., 2015a). As a result, a deterministic heuristic –which is extremely fast in execution, even for large-scale optimization problems– is extended into a probabilistic algorithm while keeping its logic to solve a specific problem. Examples of skewed probability distributions are the geometric and triangular distributions. While the triangular distribution is parameter-less, the geometric distribution is controlled by a single parameter $\beta \in [0, 1]$, which controls the level at which the selection probabilities decrease along with the sorted list. According to Grasas et al. (2017), the selection of a particular distribution relies on its capability to bias the probabilities of selecting candidate elements, defined by a bias function, which assigns a different weight, based on some criteria, to each element in the list.

Usually, the biased-random behavior is introduced at the selection step in the construction phase of the algorithm: instead of restricting the list of candidates, like in the GRASP, a different probability of being selected is assigned to each potential candidate in the list, which is sorted according to the desired criteria, e.g., the incremental cost. Therefore, the elements at the top of the list are more likely to be selected and, therefore, different executions of the algorithm are able to generate different good solutions. The advantages of the BR include the improvement of the heuristics and metaheuristics behavior in a fast and natural way, apart from possibly generating a set of alternative promising solutions, being some of them better than that one generated by the original deterministic heuristic.

Originally, Juan et al. (2010) were the first authors who employed skewed probability distributions to bias the savings heuristic for solving VRPs, inspired by the works of Faulin and Juan (2008) and Faulin et al. (2008a), which combined random sampling with heuristics. Since then, the use of BR techniques has been employed in solving different COPs in areas such as transportation (e.g., Dominguez et al. (2016a), Calvet et al. (2016a), Dominguez et al. (2016b)), and Belloso et al. (2019)), scheduling (e.g., Martin et al. (2016) and Ferone et al. (2020)), or facility location (e.g., De Armas et al. (2017) and Estrada-Moreno et al. (2019)). Another class of BR algorithms, known as biased random-key genetic algorithms (BRKGA), was introduced by Gonçalves and Resende (2011) for solving combinatorial optimization

problems. The idea behind these algorithms is to bias the selection of parents for mating. The use of BRKGA has been recently addressed for solving a range of scheduling problems (Brandão et al., 2015; Brandão et al., 2017; Homayouni et al., 2019; Kong et al., 2020).

2.4 Simheuristics

Metaheuristics and BR algorithms have been largely employed to solve COPs in the last decade. However, apart from proving that their use is quite satisfactory in solving these problems, the metaheuristics were designed under deterministic premises of the problems, and today, deterministic assumptions are becoming less acceptable in an increasingly complex world where everything is fastly changing.

Real-life based problems are often fraught with uncertainty. In the T&L context, for instance, processing times, demand, travel times, customers, etc., are commonly highly influenced by some level of uncertainty. A way to naturally extend these heuristic algorithms to deal with stochastic problems is through combining their effectiveness with the use of simulation techniques. In this way, simheuristics approaches combine the use of simulation techniques with heuristics or metaheuristics methodologies to cope with stochastic COPs (Juan et al., 2015a). Since real-life problems are mostly depending on uncertainties, the use of simheuristics can be seen as a promising approach to solve them.

Since their proposal in literature, simheuristics have gained notoriety and proven to be an efficient approach to solve stochastic problems (Rabe et al., 2020). The core of a simheuristic is based on the assumption that, in scenarios with low or moderate uncertainty, high-quality solutions for a deterministic COP are also likely to be high-quality solutions for its corresponding stochastic version. However, the best solution for the deterministic COP is not mandatorily the best solution for the stochastic variant.

Generally speaking, the resolution of a given stochastic COP instance through a simheuristic approach is performed by considering its deterministic counterpart: all the random variables are replaced by their expected values –i.e., the deterministic and fixed values, that can be seen as ‘optimistic’ values under ‘ideal’ conditions. This deterministic COP is iteratively resolved by a heuristic-driven methodology, generating a set of high-quality solutions to the deterministic COP. On every high-quality solution, a short simulation is performed to estimate its quality under stochasticity. In other words, each simulation process replaces the expected values, set previously, with the stochastic ones. Accordingly, this solution is classified (ranked) according to the stochastic problem. This process is repeated until a stop criterion is achieved. Posteriorly, a long simulation is applied to the list of elite solutions. This set of elite solutions might be re-classified for the stochastic problem, according to the long simulation process, which provides more accurate results. Lastly, a risk/reliability analysis criteria can be included in the decision-making process, by addressing expected solution failures under stochasticity, for instance. Moreover, these final simulations can also be used to obtain additional information about the probability distribution of the quality of each solution. This process is depicted in Figure 2.2.

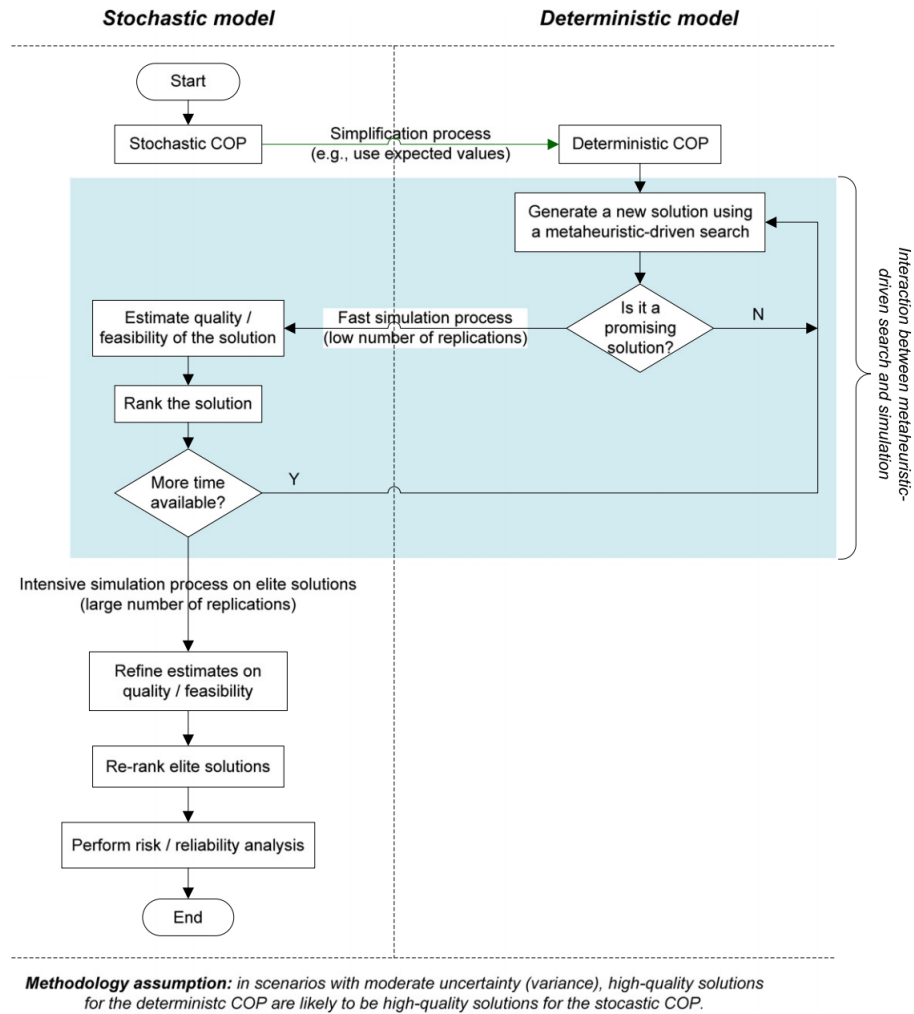


Figure 2.2: General Scheme of Simheuristics for Solving Stochastic COPs.
Source: Juan et al. (2015a).

As an extension of heuristic and metaheuristics approaches, simheuristics do not guarantee the generation of optimal solutions. However, according to Chica et al. (2017), they are capable to provide decision-makers with risk or reliability analysis criteria regarding the stochastic solutions during the assessment of alternative high-quality solutions to the optimization problem. In addition, simheuristics are relatively easy to implement, scalable, capable to cope with uncertainties and provide good solutions in reasonably short computational time. Moreover, different probability distributions can be employed to add a random biased behavior to classical heuristics through the hypothesis that each one has its characteristics and may perform better in a proper use scenario.

The simheuristics have been applied to solve a wide set of stochastic COPs in different fields, such as permutation flow-shop (e.g., Hatami et al. (2018) and Villarinho et al. (2021)), facility location problems (e.g., Gruler et al. (2020) and De Armas et al. (2017)), inventory routing problems (e.g., Gruler et al. (2018), Onggo et al. (2019), Gruler et al. (2020), and Raba et al. (2020)), telecommunication networks (e.g., Cabrera et al. (2014) and Alvarez Fernandez et al. (2021)) and finances (e.g., Panadero et al. (2020b)). Interestingly enough, simheuristic

approaches have been developed to cope with behavioural models in air combat behaviour (Lam et al., 2019), assembly lines Lopes et al. (2020), and iron ore crusher circuits (Santos et al., 2020).

2.5 Agile Optimization

Despite being able to generate feasible and efficient solutions in a very short time, heuristics and BR heuristics capabilities are often underused by generating only a single solution during their lifetime, being them frequently embedded in a multi-start framework, which is a sequential and iterative approach. However, although being able to explore different solutions and return the best-found at the end of this process, the sequential particularity of the multi-start might transform this resulting strategy into a slow solving methodology.

In order to deal with such particularity, agile optimization (AO) takes advantage of combining two powerful approaches from both parallel computing (Malapert et al., 2016) and biased-randomization of heuristics (Grasas et al., 2017). For being extremely fast in execution, easily parallelizable, flexible, and requiring the fine-tuning, in many cases, of only a single parameter, this combination, which represents a new optimization perspective, allows the finding of reasonably high-quality solutions for a range of large-scale and *NP-hard* optimization problems in real-time.

In particular, the idea behind this technique is to run, in parallel, several hundred or even thousands of threads, being each one execution of a BR heuristic. As a result, several alternative solutions are generated in the same wall-clock time as the one employed by the original heuristic for running a single solution. Consequently, a set of different, but feasible and of good quality, solutions is provided –some of them outperforming the one generated by the original heuristic– and the best solution is chosen. Another advantage of this strategy refers to the possibility of providing decision-makers with different solutions whose different characteristics might be attractive for them.

Particularly, AO algorithms are suitable to deal with problems that require real-time decision-making, for instance, systems that require a re-optimization whenever a new piece of information should be incorporated into the model or systems that involve risky situations. This concept is also necessary when dealing with dynamic systems (e.g., traffic, vehicle location, unexpected demands, disruptions, etc.), where the environmental conditions are continuously changing, and re-optimization of the system is required every few minutes or even seconds.

Figure 2.3 represents the general idea of the AO framework in which n concurrent executions of a BR heuristic ($BRH_i, i \in \{1, \dots, n\}$) are run in parallel. Each thread is smoothed by applying a $\beta \in [0, 1]$. As result, the best-found solution, among those several alternative solutions generated at the same time, is returned.

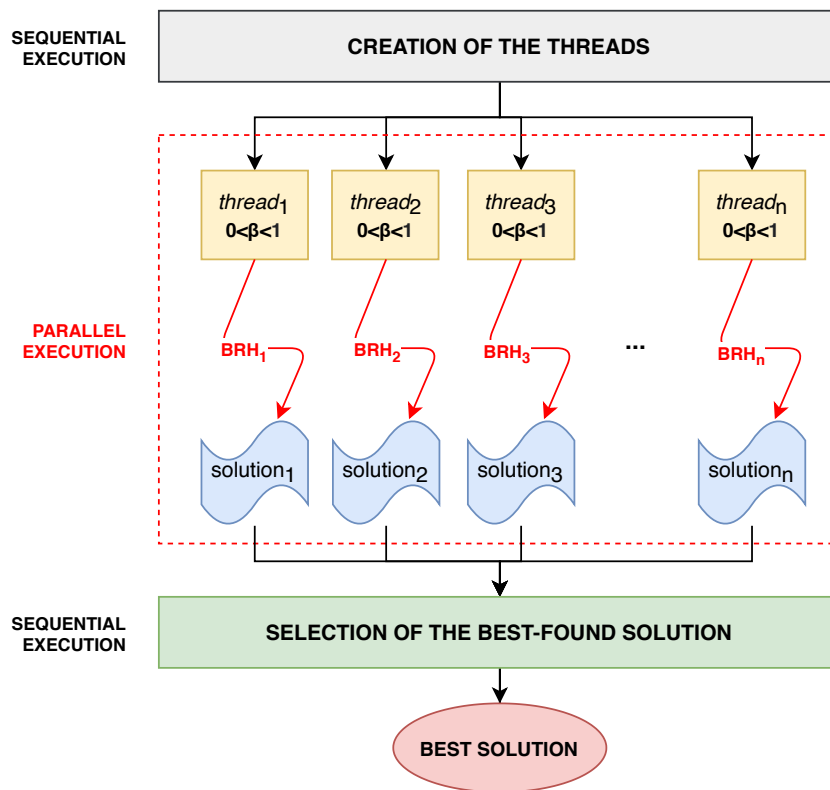


Figure 2.3: The general idea of the Agile Optimization framework: to run several executions of a biased-randomized heuristic in a parallel way.

2.6 Conclusions

Five classes of optimization techniques have been addressed and described so far: (i) heuristics; (ii) metaheuristics; (iii) biased-randomized heuristics; (iv) simheuristics; and (v) agile optimization algorithms. As discussed at the beginning of this chapter, the use of each solution method for solving a COP relies not only on its main characteristics but also on its ability to adapt to a particular environment, exploit the structure of the problem, and escape from local optima.

In Figure 2.4, the particularities of multiple analytical methodologies is presented and compared in terms of the following dimensions: (i) capacity to provide optimal values (exact methods); (ii) computational time required (both heuristics and agile algorithms show the highest speed levels, offering real-time solutions); (iii) flexibility to model real-life situations (simulation); (iv) capacity to deal with uncertainty scenarios (simulation and simheuristics); (v) capacity to deal with large-scale problems (heuristics, metaheuristics, and agile algorithms); and (vi) capacity to deal with dynamic environments (learnheuristics, heuristics, and agile algorithms). These comparisons are performed considering a scale from 1 (low performance) to 5 (high performance).

Although exact methods can provide optimal solutions, their use is often restricted due to the complexity of the problem to be tackled, the magnitude of the instances, and the required computational time. On the other hand, (BR)-heuristics and metaheuristics are faster

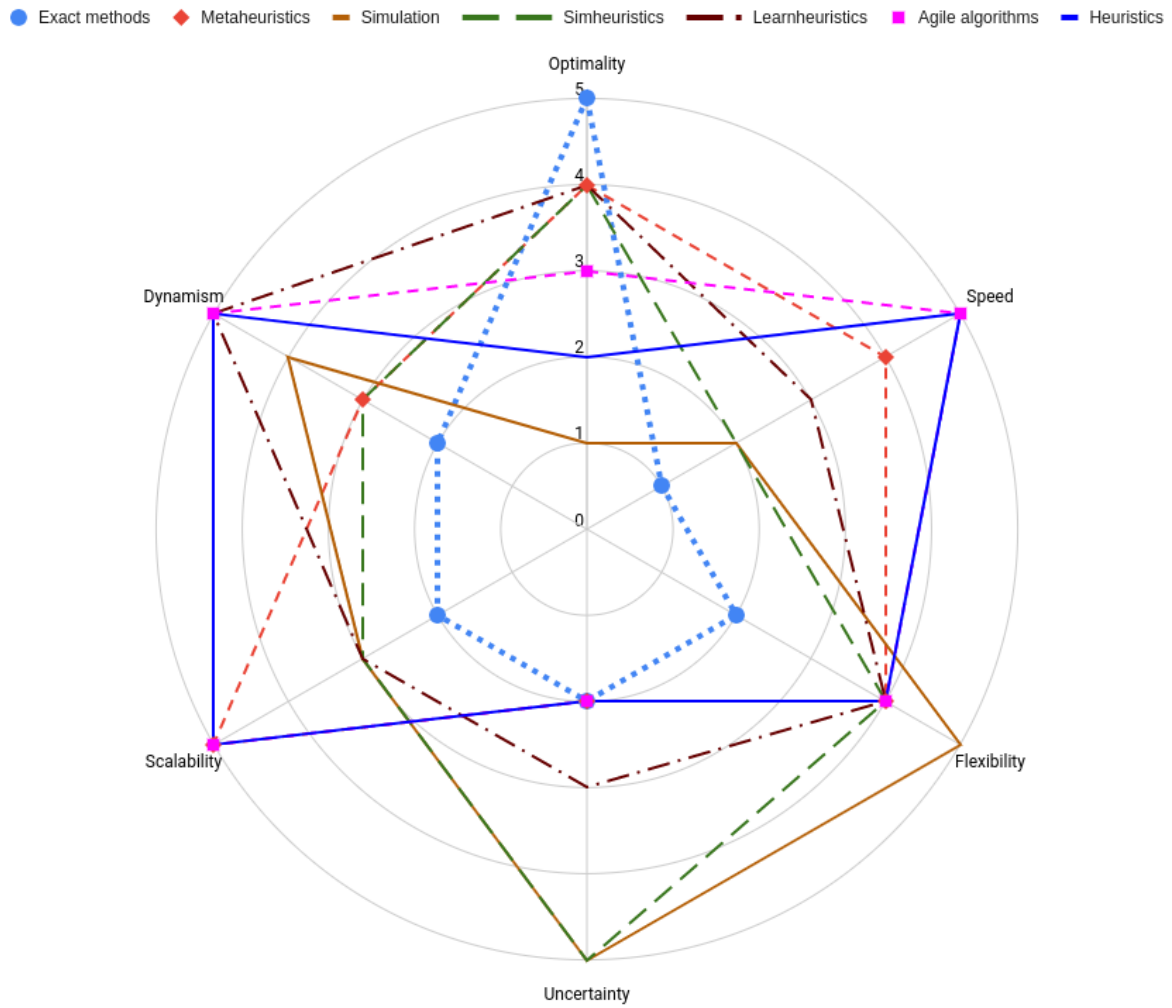


Figure 2.4: Multi-dimensional comparison of different analytical approaches.
Source: Martins et al. (2020b).

than exact approaches, are able to cope with large-scale problems, but they do not ensure the finding of optimal solutions. However, metaheuristics can find high-quality (or even optimal, or near-optimal) solutions for a vast of COPs. Simheuristics algorithms are an excellent strategy to deal with uncertainty, but they require additional computation effort when compared with heuristics and AO algorithms, due to simulation processes. BR and AO algorithms are flexible, scalable, and capable to deal with dynamism, but they fail in dealing with uncertainty and optimality. Therefore, there is not an ‘optimal’ method that can tackle all these T&L problems that arise nowadays, being the use of each solving methodology strongly dependent on the problem structure, its characterization, and also on the hardware capabilities.

Chapter 3

Applications in Transportation

This chapter ¹ studies the emerging Omnichannel Vehicle Routing Problem (OCVRP) and the Vehicle Routing Problem with Optional Backhauls (VRPOB). On one the one hand, the OCVRP is a two-echelon VRP, found in many omnichannel retailing systems, which combines two delivery levels, in which the first one addresses the delivery from the depot to intermediate facilities, while the second level regards the delivery from these intermediate facilities to final customers, i.e., the last-mile delivery. In this chapter, two variants of this problem are addressed: the deterministic and stochastic OCVRP. The following solution methods are proposed to solve the OCVRP: (i) a savings-based heuristic; (ii) a biased-randomized savings-based heuristic; (iii) a multi-start approach; and (iv) a simheuristic approach. On the other hand, the VRPOB is an extension of the classical VRP in which returnable transport items are optionally collected by the vehicles previously employed to make deliveries. To solve the VRPOB, a biased-randomized iterated local search is proposed.

3.1 The Omnichannel Vehicle Routing Problem

Unlike brick-and-mortar stores, where salespeople are available to support and help customers to make their purchases, recent advances in Information and Communication Technologies (ICT) and the popularization of mobile devices with access to the Internet have promoted the use of different shopping channels. The online channel is an example that has emerged as a competitive marketing channel to that of traditional retail centers, transforming e-commerce into a global trend and an important tool for every business worldwide (Abdulkader et al., 2018). With e-commerce, customers are immersed in an environment of a plethora of information, opinions, and access to a vast combined supply of stock, which,

¹The contents of this chapter are based on the following works:

- **Martins, L. C.**; Bayliss, C.; Juan, A. A.; Panadero, J.; Marmol, M. (2020): [A savings-based heuristic for solving the omnichannel vehicle routing problem with pick-up and delivery](#). *Transportation Research Procedia*, 47, 83-90.
- **Martins, L. C.**; Bayliss, C.; Copado-Méndez, P. J.; Panadero, J.; Juan, A. A. (2020): [A Simheuristic Algorithm for Solving the Stochastic Omnichannel Vehicle Routing Problem with Pick-up and Delivery](#). *Algorithms*, 13(9), 237.
- Bayliss, C.; **Martins, L. C.**; Juan, A. A. (2020): [A two-phase local search with a discrete-event heuristic for the omnichannel vehicle routing problem](#). *Computers & Industrial Engineering*, 148, 106695.
- Londoño, J. C.; Tordecilla, R.; **Martins, L. C.**; Juan, A. A. (2020): [A biased-randomized iterated local search for the vehicle routing problem with optional backhauls](#). *TOP*, 1-30.

together, allows them to browse through different stores in an online environment. Although some experts predicted that online shops would kill the physical ones, the truth is that they coexist and have transformed the way customers shop nowadays (Gallino and Moreno, 2014).

The use of a variety of shopping channels is referred to as *omnichannel retailing*, where, instead of having only the single option of physically visiting a store to buy products, for instance, consumers can also buy them via online shopping to be delivered to their homes. In other words, omnichannel retailing refers to the use of a variety of marketing channels and therefore delivery channels to fulfill customer orders and provide them a seamless experience (Chopra, 2016). Customers switch among channels, among retailers, and among devices for different reasons and different valuing factors, including brand perception, atmosphere, price, availability of stock, convenience or personalization (Agnihotri, 2015). Although this channel integration provides consumers with a seamless shopping experience (Zhang et al., 2018), it also requires the design of integrated distribution systems. For instance, the same fleet of vehicles should be able to replenish the stocks of retail stores and, at the same time, service the product demands placed by online customers. Such integration of tasks introduces complex precedence constraints into vehicle route planning, since: (i) customer orders need to be picked up from retail centers before they can be delivered; and (ii) those retail centers need to be replenished simultaneously. According to Hübner et al. (2016a), the need for improving the delivery system in this retailing environment includes the development and optimization of delivery modes and the increase of the delivery speed—since most customers want to receive the delivery up to a maximum of two days after placing the order.

The well-known Vehicle Routing Problem (VRP) is an operational decision that aims to design cargo vehicle routes with minimum transportation costs in order to distribute goods between depots and a set of final customers during a planning period without causing stock-outs at any of the customers (Crainic and Laporte, 2012). Some VRP variants also consider facility location costs (Quintero-Araujo et al., 2019). In the case of the OCVRP, this integrated problem is presented as a variant of the classical VRP and can be treated as an integration of two of its different variants: the Capacitated VRP (CVRP) and the Pickup and Delivery Problem (PDP). The omnichannel VRP considered in this work can be considered as an integrated Two-Echelon VRP (2E-VRP), since retailers and online consumers correspond to the customers of two distinct layers of the supply chain.

The VRP in omnichannel distribution systems was first introduced by Abdulkader et al. (2018) and it is depicted in Figure 3.1. In this problem, a single distribution center supplies a group of retailer stores, which represents the first-echelon distribution level. In turn, this set of retail centers serve a set of online customers, i.e., the second-echelon. A single fleet of vehicles is employed in both delivery levels in order to reduce transportation costs. Apart from reducing operating costs and improving supply chain competitiveness, the optimization of this two-echelon problem holds the potential to improve customer service levels and to enhance the on-time delivery of customer orders (Cheng et al., 2015).

Since the classical VRP and PDP are both *NP-hard* problems (Lenstra and Kan, 1981b;

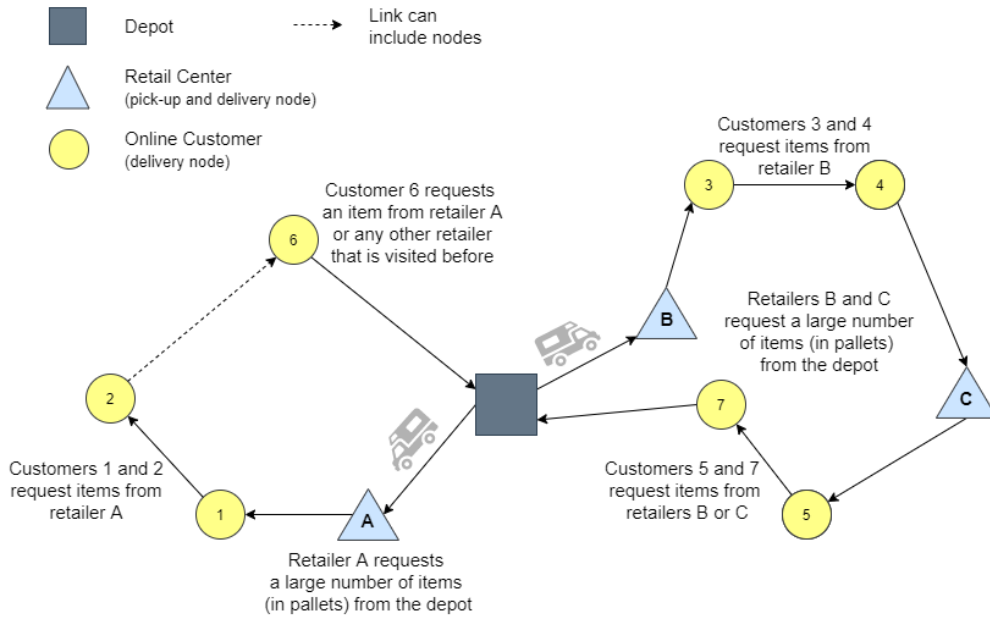


Figure 3.1: A general schema of the OCVRP.

Savelsbergh and Sol, 1995), the combination of these two variants is certainly *NP-hard*. In this way, in this chapter, a simple and fast *savings-based heuristic* is firstly proposed to solve the OCVRP. This heuristic is designed under specific characteristics of the considered problem and guided by smart decisions to generate feasible solutions in a short computational time and to avoid several repairing processes when an infeasible solution is generated. It contrasts with the existing literature, which proposes a heuristic methodology that spends a large amount of time on corrective actions to generate feasible results. This heuristic is, posteriorly, extended into a *biased-randomized savings-based heuristic*, which is embedded into a multi-start approach by incorporating a skewed probability distribution to smooth the original deterministic behavior, transforming it into a probabilistic one.

The previous solution methods are designed to cope with the deterministic variant of the OCVRP. VRPs and PDPs are frequently considered in the literature as deterministic problems, where customers' demands and travel times are constant values. However, in real life, it is frequent that both demands and travel times are exposed to some degree of uncertainty. In these scenarios, it is more accurate to model these constant variables as random variables. Therefore, the OCVRP formulation is extended into a stochastic variant by replacing the deterministic travel times with stochastic ones.

In order to deal with the SOCVRP, the proposed MS approach is combined with Monte Carlo simulation (MCS) and extended into a *simheuristic* algorithm (Juan et al., 2018). Simulation-optimization methods and simheuristics, in particular, allow us to properly deal with stochastic versions of COPs (Gonzalez-Martin et al., 2018; Gruler et al., 2017), such as the SOCVRP, where travel times are modeled as random variables. The simheuristic approach is also employed to measure the 'reliability' level of the proposed distribution plan, i.e., to measure the probability that the plan can be deployed, without any route failure, in a realistic scenario under uncertain travel times. To the best of our knowledge, this is the first time that a

stochastic version of the OCVRP has been considered in the scientific literature.

3.1.1 Literature Review

As mentioned, the OCVRP can be seen as a combination of the VRP and a PDP. The classical VRP, originally proposed by Dantzig and Ramser (1959), has been extensively studied by practitioners and academics due to its wide applications in several areas. The VRP belongs to a set of *NP-hard* COPs (Lenstra and Kan, 1981b). Therefore, the use of exact algorithms is very limited and efficient only for solving small-sized VRP instances. Mostly, these exact approaches are based on the combination of column and cut generation algorithms (Laporte et al., 1986; Fukasawa et al., 2006). On the other hand, approximate algorithms, such as metaheuristics, are frequently very efficient for solving large-sized instances of COPs. Several metaheuristics have been proposed to solve the VRP, which include tabu search (TS), genetic algorithms (GAs), ant colony optimization (ACO), and some hybrid methodologies (Barbarosoglu and Ozgur, 1999; Nazif and Lee, 2012; Ezzatneshan, 2010; Lin et al., 2009). PDPs have also been studied for more than 30 years. These problems incorporate some route order dependencies in which some nodes should be visited before others in order to transfer inventory between them. Similar to the VRP, the PDP is also an *NP-hard* problem (Savelsbergh and Sol, 1995), and some exact methodologies, based on branch-and-cut and branch-and-cut-and-price algorithms, have been developed to optimally solve small-sized PDP instances (Lu and Dessouky, 2004; Ropke and Cordeau, 2009). Moreover, several heuristics and metaheuristics algorithms have been developed to solve the PDP and some of its variants. We highlight the use of TS (Nanry and Barnes, 2000), GA (Pankratz, 2005), large neighborhood search heuristics (LNS) (Ropke and Pisinger, 2006), adaptive LNS (ALNS) (Li et al., 2016; Ghilas et al., 2016), particle swarm optimization (PSO) (Ai and Kachitvichyanukul, 2009), and greedy clustering methods (Nadizadeh and Kafash, 2019). A literature review and classification of PDPs is presented by Berbeglia et al. (2007).

One of the first studies concerning two-level routing problems was presented by Jacobsen and Madsen (1980) in order to solve the daily distribution of newspapers. However, only decades later, Crainic et al. (2004) introduced the 2E-VRP, motivated by the use of intermediate facilities to redistribute goods where large trucks were not able to circulate due to physical limitations of the streets. Consequently, the use of these intermediate facilities reduced the use of large vehicles by up to 72%. Years later, a study on the relationships between customer distribution, system layout, and the associated costs of the distribution process for two-echelon distribution systems was provided by Crainic et al. (2010). They measured and analyzed the impact of the number of customers, the quantity and location of intermediate facilities, the customer distribution, and the relationship between the first and second level costs on the total cost of distribution. The authors concluded that opening facilities reduce the global cost until a minimum cost is reached, and from that minimum, adding new ones increases the global cost. Different approximation methods have been proposed to solve the 2E-VRP. Hemmelmayr et al. (2012) proposed an ALNS for solving the 2E-VRP. The authors developed new search operators based on the problem structure and were able to outperform existing results from the literature. Additionally, a new data set

of large instances was proposed for the problem. More recent studies solve the 2E-VRP by using hybrid methodologies. Crainic et al. (2013) presented a combination of GRASP with path-relinking for solving the 2E-VRP while a combination of GRASP with a VND has been presented by Zeng et al. (2014). Both hybrid methodologies were able to improve existing results in the literature. Recently, the 2E-VRP has been studied by introducing electric vehicles for the second-echelon deliveries (Breunig et al., 2019). In this case, an LNS metaheuristic was proposed to solve large-scale instances of the problem.

Another problem to which the OCVRP can be related is the VRP with pick-up and delivery and, in particular, the VRP with simultaneous pick-up and delivery (VRPSPD). In the case of the OCVRP, some of the deliveries depend on earlier pickups in the route, thus introducing additional precedence constraints. Due to the complexities of such problems, the vast majority of previous work in this area is concerned with the application of heuristic methodologies. The VRPSPD was first tackled by Min (1989) for solving a real-life problem. The authors proposed a three-stage heuristic in order to minimize the total travel time of the routes. Firstly, the customers were clustered complying with the vehicle capacity per group. In the next step, one vehicle was assigned to each cluster, and finally, a traveling salesman problem (TSP) was solved for each group. From this work, several heuristics and metaheuristics have been proposed to solve the VRPSPD. Crispim and Brandão (2005) and Bianchessi and Righini (2007) have considered several TS approaches with large complex neighborhoods and adaptive tabu list structures. Montane and Galvao (2006) also considered a TS and compare the performance of three inter-route neighborhoods (relocation, interchange, and crossover) combined with one intra-route neighbor or *2-opt*. They find that a combination of all three inter-route neighborhoods is better in most but not all problem instances. Mu et al. (2016a) consider a parallel implementation of SA based on modeling search trajectories as multiple Markov chains with an integrated synchronous and asynchronous scheme for sharing new best solutions between threads. Tasan and Gen (2012) consider a GA approach to the same problem using a permutation solution representation, in which solutions are evaluated by visiting as many nodes as possible in the string with the same vehicle before deploying a new vehicle starting from the depot. This decoding approach guarantees solution feasibility at the risk of destroying any general schema that may have previously been identified by the algorithm. Ai and Kachitvichyanukul (2009) consider a PSO approach to the same problem in which a particle's position encodes a solution. In particular, there is a dimension for each customer node, and the position in the customer's dimension determines their delivery priority. In each iteration, all particle velocities are updated according to inertial, cognitive, and social criteria. Inertia acts to keep a particle moving in the same direction, while cognitive criteria encourage particles to travel towards the position of the best-known solution. Finally, social criteria encourage the particle to move in the same direction as nearby particles. Their approach finds some new best-known solutions. More hybrid methodologies, such as ACO with local search (Gajpal and Abad, 2009; Çatay, 2010), and PSO combined with local search and VND (Goksal et al., 2013; Ai and Kachitvichyanukul, 2009), respectively, have been addressed in the literature. Finally, a parallel methodology based on SA has been considered to solve the problem (Mu

et al., 2016b).

An integration of 2E-VRP with simultaneous pick-up and delivery was addressed by Belgin et al. (2018), who proposed a hybrid heuristic based on VND and local search to solve medium and large-sized instances of the problem. In this work, the same fleet of homogeneous vehicles is employed to serve both delivery levels. However, a different vehicle capacity is imposed for each service level. Despite being a similar problem to that one addressed in this section, the OCVRP problem is characterized by different constraints and decisions in the routing sequence, including the shared use of vehicle capacity for both delivery levels. To the best of our knowledge, Abdulkader et al. (2018) were the first authors to address the OCVRP as a combination of the VRP and the PDP in an omnichannel retailing context. For solving this novel integrated problem, the authors proposed a two-phase heuristic, based on: (i) inserting consumers into retailers routes and on correcting infeasible solutions; and (ii) on joining the routes through the maximum-savings criterion, i.e., the Clarke & Wright Savings (CWS) heuristic (Clarke and Wright, 1964). Apart from this two-phase heuristic, a multi-ant colony algorithm (MAC) was proposed. A complete set of instances have been generated to test their methodologies, and the MAC outperformed the heuristic's performance. A complete set of small- and large-sized instances was generated. The authors proved that the use of an integrated distribution system is able to reduce transportation costs by up to 45% when compared with the separated one.

The expansion of information technologies has made possible the rise of different shopping channels, as a result customer behaviour has evolved accordingly (Mosquera et al., 2017). Several authors have addressed the transition from multichannel to omnichannel retailing (Verhoef et al., 2015; Hübner et al., 2016b; Mena et al., 2016). Although customers use physical and online stores when purchasing products, the customer experience is different in each channel. While the online and offline channels are treated as separate businesses in multichannel retailing, the use of both channels in an omnichannel environment provides consumers with a seamless experience (Heitz-Spahn, 2013). A study of effective and efficient operations within an omnichannel distribution system is addressed by Hübner et al. (2016a). From this study, the authors concluded that excellence in omnichannel distribution is achieved by expanding delivery modes, increasing delivery speed, and service levels. A complete study of multiple-channel retailing categories has been provided by Beck and Rygl (2015), who offers concise definitions on the concepts of multi, cross, and omnichannel distribution.

Despite the fact that travel times are stochastic in most real-life transportation activities, in the past only deterministic versions of the OCVRP have been considered. Hence, this Section goes one step further by considering random travel times and proposing a simheuristic algorithm to solve the stochastic version of the OCVRP.

3.1.2 Problem Definition

As previously mentioned, the OCVRP addressed in this thesis can be considered as a combination of the VRP and the PDP, being its main novelty, in comparison to other VRPs, is the

precedence constraints regarding the collection of customer orders from retailers and subsequent delivery to customers. In the OCVRP, a single fleet of vehicles, which is available at a central and single depot at the start time t_0 , is employed to simultaneously perform three different operations: (i) bulk deliveries to retail stores from a main central depot; (ii) the pick-up of online customer orders from these retail stores; and finally, (iii) the deliver of online customers' orders, which are geographically distributed in the network. In other words, each vehicle departs from the depot with enough loaded demand to service the retail stores in the route. Once at the retail centers, the demand for these unprocessed items is delivered and the products required by the customers are picked up. These products are later delivered within the same route. Therefore, the vehicle responsible for a particular online order must first visit a retail store with that product in stock.

Usually, retail stores must be supplied from the depot with a large number of unprocessed items, which are packed and frequently measured in terms of the number of pallets. In contrast to retail stores' demand, online customer orders are of negligible size but require processing at a physical retail store before their delivery. Hence, due to the additional handling and/or packing, demands from online customers cannot be directly delivered from the depot. Different types of products are available to purchase via the online channel at each retail store. Likewise, each retail store has a limited inventory of processed products that can be delivered to online customers. Since the products required by the customers are considered to be negligible in load, they do not affect the vehicle capacity.

The available processed inventory and bulk demand for each retail store are known in advance, and each delivery, either for retail stores or individual customers, requires a drop time τ and the vehicle returns to the depot when all the deliveries are concluded. It is assumed that bulk inventory does not contribute to a retailer's available inventory since bulk deliveries cannot be processed within the drop time of delivery. As a simplifying assumption, each online customer orders a single product. When more than one item is ordered, the consumer is replicated in the same location as a 'virtual' consumer, and each product is delivered separately. According to Abdulkader et al. (2018), this strategy of considering single-item orders by consumers guarantees the solution feasibility, apart from minimizing the distribution cost.

3.1.2.1 Deterministic Omnichannel Vehicle Routing Problem

The distribution network of the OCVRP can be defined as a directed graph $G = (N, A)$, with a set of vertices $N = \{0, 1, \dots, |RC| + |O|\}$, in which node 0 represents the depot, RC is the set of retail centers, and O is the set of online customers. The set N comprises both sets RC and O . The set of arcs is $A = \{(ij) \in N^2\}$. For the deterministic OCVRP, the travel time to traverse an edge (i, j) is given by the Euclidean distance which is calculated according to the corresponding coordinates of its respective nodes. The objective when solving this problem is to minimize the total cost of the vehicle routes such that:

1. Every route starts and ends at the depot.
2. The routes do not exceed the maximum tour length.

3. Every node $i \in N \setminus 0$ is visited by only one vehicle and only once.
4. The total delivery demand from the depot to the retail stores does not exceed the vehicle capacity.
5. The total demand of a customer to be served from the retail stores with a certain product cannot exceed the available inventory of this product at the retail centers.
6. The retail store determined to satisfy the consumer's demand must be visited before the consumer and by the same vehicle.

Two alternative mathematical formulations of the deterministic OCVRP can be found in Abdulkader et al. (2018) and Bayliss et al. (2020b). Figure 3.2 provides a hypothetical small solution example, composed of one route with a single central depot (0), one retail center, two customers, and a single type product. In this case, the maximum route time, T , is 20 time units, the vehicle capacity, C , is 50 weight units, and the drop time, τ , is 1 time unit. Each node is characterized by the tuples (id, vl, t, np) and $[R, S]$, where: id is the node identifier; vl is the current vehicle load after visiting node id ; t is the current time after visiting node id ; and np is the number of remaining products available in the vehicle after visiting node id . Variables R and S indicate the raw and processed demand of each node, respectively. In this example, the vehicle departs from the depot 0, which supplies the vehicle with 50 units of raw product, at time 0, with no picked up orders. At the retail store 1, which demands 50 raw product units and provides 2 processed units, the vehicle drops-off the required demand and picks up the items to be delivered to the costumers. The vehicle departs at time 6 (distance plus drop time) to service customer 2. Now, the vehicle drops off the product required by the current customer and travels to the next customer at time 11. Similarly, the vehicle drops off the product required by customer 3 and returns to the depot at time 14 completely empty. The solution cost is 14 distance units and its corresponding tour length is 17 time units –which accounts for both the travel time and the drop time at each delivery node.

3.1.2.2 Stochastic Omnichannel Vehicle Routing Problem

As an extension of the deterministic OCVRP, the stochastic OCVRP can be defined by the same directed graph $G = (V, A)$, where the set of vertices N comprised both the customers, retail centers, and central depot. However, the previous formulation is extended by considering the case in which edge-traversal times are stochastic, in contrast to the original problem addressed in the literature. In this case, the time to traverse an edge (i, j) is defined as $T_{ij} = t_{ij} + D_{ij}$ where t_{ij} is the deterministic edge-traversal time (which represents the edge-traversal time under 'ideal' traffic conditions), D_{ij} is a log-normally distributed delay term, and T_{ij} denotes the distribution of edge-traversal times. The introduction of this uncertainty into the classic OCVRP transforms it into a probabilistic problem. Therefore, apart from considering the same set of constraints (1)-(6) as the deterministic problem, the objective of the stochastic OCVRP is to minimize the total travel cost of the vehicle routes such that the routes must have a reliability level greater than a user-set parameter, R_{min} . The

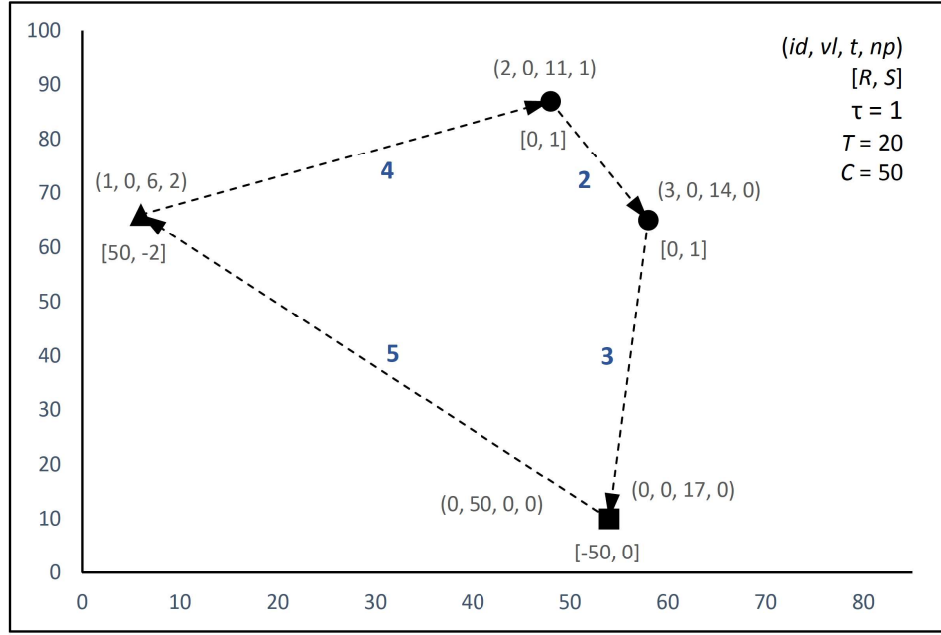


Figure 3.2: A practical example of the OCVRP distribution system.

reliability level of a set of routes is defined as the probability that all routes are completed within the maximum time limit of T_{max} .

3.1.2.2.1 Mathematical Model As introduced, alternative mathematical formulations of the deterministic OCVRP were presented by Abdulkader et al. (2018) and Bayliss et al. (2020b). Differently from their formulations, a stochastic objective function and a probabilistic constraint regarding route completion reliability are considered for the SOCVRP. Let x_{fkij} be a binary decision variable indicating whether or not vehicle f in the fleet F traverses edge ij in the k^{th} node visit in its route. The stochastic objective function is given in Equation (3.1), where D_{ij} is the stochastic delay associated with traversing edge ij .

$$\min E \left(\sum_{f \in F} \sum_{k \in K} \sum_{i \in V} \sum_{j \in V} x_{fkij} (t_{ij} + D_{ij}) \right) \quad (3.1)$$

The probabilistic route completion reliability constraint is given in Equation (3.2).

$$R_{min} \leq P \left(\sum_{i \in V} \sum_{j \in N} \sum_{k \in K} (t_{ij} + \tau + D_{ij}) x_{fkij} + \sum_{i \in V} \sum_{k \in K} (t_{i0} + D_{i0}) x_{fki0} \leq T_{max}, \forall f \in F \right) \quad (3.2)$$

Equation (3.2) also accounts for the drop times (τ) that are required when performing deliveries and pick-ups. The vehicle capacity constraint, regarding the retailer demands that can be satisfied, is expressed by Equation (3.3), where OP_j denotes the number of ordered product units of node j (which is zero for customer nodes) and H denotes the vehicle capacity.

$$\sum_{i \in N} \sum_{j \in V} \sum_{l=1}^k x_{flij} OP_j \leq H, \forall f \in F, \forall k \in K \quad (3.3)$$

The customer order precedence constraints are expressed by Equation (3.4), where OD_{jq} denotes the number of items of type $q \in Q$ ordered by node j (which is zero for retailer nodes) and P_{jq} denotes the number of items picked-up of type $q \in Q$ at node j (which is zero for customer nodes).

$$\sum_{i \in V} \sum_{j \in V} \sum_{l=1}^k x_{flij} (P_{jq} - OD_{jq}) \geq 0, \forall f \in F, \forall k \in K, \forall q \in Q \quad (3.4)$$

3.1.3 Solution Method: From a Heuristic to a Simheuristic Approach

To solve the OCVRP, a savings-based heuristic is proposed (Martins et al., 2020a). This heuristic is based on the well-known CWS heuristic (Clarke and Wright, 1964) and is composed of four stages. This heuristic, which is completely deterministic, is posteriorly extended into a biased-randomized heuristic, and, finally, to a multi-start algorithm. A local search procedure, which incorporates the use of a hash map data structure that keeps the best-found route sequence for each set of nodes (Juan et al., 2011), is applied to search for locally optimal solutions.

3.1.3.1 The Savings-Based Heuristic

As introduced, the savings-based heuristic, henceforth named *LH*, is based on greedy decisions and designed to solve the deterministic variant of the OCVRP. Pseudocode 5 describes, in details, the *LH*, which is composed of the following three stages:

1. In the first stage (line 1), a dummy solution is created, which is composed of a set of ‘dummy’ routes. Each dummy route is designed to serve one node $i \in N$, which can be either a consumer or a retail store. The route departs from the depot, travels to the node, and then returns to the depot.
2. The second stage of *LH* (lines 4-19), named as CWS_1 , is also presented in the Flowchart 3.3, where the dummy routes from stage 1 are merged using a maximum-savings criterion (Clarke and Wright, 1964). Each box from Flowchart 3.3 is numbered to aid the following explanations.

Initially, a savings list (SL) is constructed (line 2, step 1). This list considers all possible pairs of nodes (i.e., edges) from the problem. For each edge $\{i, j\}$, the corresponding savings value is calculated as $s_{ij} = t_{0i} + t_{j0} - t_{ij}$, where t_{ij} represents the deterministic travel time between nodes i and j . That is, candidate solutions are generated using ‘ideal’ traffic conditions for the edge-traversal times.

Initially, all edges from SL are eligible. This eligibility is related to constraints regarding the retail centers’ inventory. The list is sorted in descending order of the savings value (step 2), and the edge with the highest saving is selected (line 3, step 3). At this stage, the selection of edges is restricted to guarantee the assignment of a retail center

to each consumer in the problem (line 10, step 5). In other words: a route containing one single customer can only be merged with a route containing a retail center, i.e., the selection is restricted to eligible edges $\{i, j\}$, where node j is a consumer in a dummy route and i is a node in a route with a retailer that can supply consumer j . Figure 3.4a represents a merging case in which i is a retail center that can supply customer j . In Figure 3.4b, on the other hand, i is a customer in a route with a retail center that can supply customer j . These attempts at only merging routes containing at least one retail center, which can supply a single consumer, are made first in order to avoid infeasible solutions –i.e., solutions in which some customers are not assigned to any retailer. This approach of addressing solution feasibility first is based on the observation that the availability of feasibility restoring merges will only decrease as the algorithm progresses.

Based on CWS route-merging conditions (line 11, step 6), the two corresponding routes, i and j , of an edge $\{i, j\}$ (obtained in steps 4.1 and 4.2) can be merged only if: (i) nodes i and j are exterior in their respective routes (a node is exterior if it is adjacent to the depot); (ii) i and j belong to different routes; (iii) the maximum tour length is not violated; and (iv) the vehicle capacity is not violated.

The selected edge is deleted from SL (line 15, step 9) only if: (a) the corresponding merge is performed (step 7); or (b) at least one of the CWS constraints ((i)-(iv)) are violated (line 11, step 6). Otherwise, the edge becomes temporarily ineligible (line 17, step 10), but it is not removed from the list since subsequent merges might restore eligibility. This can occur when a different retail center is merged into a route, increasing the available processed inventory for subsequent consumers (line 12, step 7). For example, when selecting an edge (i, j) , the evolving route of i may have insufficient processed inventory for customer j at this time. However, if in subsequent iterations route i is merged with another route containing retailers, the evolving route of i may then be able to serve customer j .

When a merge is successfully performed (line 12, step 7), the entire SL becomes eligible (line 13, step 8), since a new inventory scenario is generated.

At the end of this stage, all the consumers are supplied by the retail centers, guaranteeing a feasible final solution. Notice that this is achieved without solving separate assignment and routing problems, as done in Abdulkader et al. (2018).

3. Finally, the third stage (lines 20-30), CWS_2 , tries to improve the solution generated in the previous step. To do that, the algorithm cycles through the reduced SL list, which includes the remaining saving edges that were discarded in the previous step, with the aim of identifying more beneficial merges. Unlike the procedure used in the CWS_1 stage, in this phase all the customers are already assigned to a retail center, so step 5 of Flowchart 3.3 and line 10 of Pseudocode 5 are not required. Hence, all edges become eligible. The process attempts all the available merging possibilities which may improve the solution. Each time a new edge is selected from the SL list, it is

removed from the list, whether the corresponding merge is performed or not— due to it violating any of the constraints (i) to (iv) (line 29). At each new iteration, the highest saving edge is selected to restart the merge process (line 21). This process is repeated until SL is empty (line 20). At the end of the procedure, a feasible solution is generated, without the necessity of repair operations.

Pseudocode 5: Savings-Based Heuristic

```

Data: set of nodes  $V$ 
1 Function LH( $V$ ):
2    $sol \leftarrow \text{createDummySolution}(V)$ 
3    $L \leftarrow \text{createSavingsList}(sol)$ 
4    $L \leftarrow \text{sort}(L)$ 
5   while there are eligible edges in  $L$  do
6      $e \leftarrow \text{selectTheFirstEligibleEdgeFromList}(L)$ 
7      $iNode \leftarrow \text{getOrigin}(e)$ 
8      $jNode \leftarrow \text{getEnd}(e)$ 
9      $iRoute \leftarrow \text{getEvolvingRouteOfNode}(iNode)$ 
10     $jRoute \leftarrow \text{getEvolvingRouteOfNode}(jNode)$ 
11    if  $jNode$  is a non-served FAL and  $iRoute$  has a PL that can serve  $jNode$  then
12      if all route-merging conditions are satisfied then
13         $sol \leftarrow \text{mergeRoutesUsingEdge}(e, iRoute, jRoute, sol)$ 
14         $\text{edgesEligibility}(L, true)$ 
15      end
16       $\text{deleteEdgeFromList}(e, L)$ 
17    else
18       $e \leftarrow \text{eligibility}(e, false)$ 
19    end
20  end
21  while there are edges in  $L$  do
22     $e \leftarrow \text{selectTheFirstEdgeFromList}(L)$ 
23     $iNode \leftarrow \text{getOrigin}(e)$ 
24     $jNode \leftarrow \text{getEnd}(e)$ 
25     $iRoute \leftarrow \text{getEvolvingRouteOfNode}(iNode)$ 
26     $jRoute \leftarrow \text{getEvolvingRouteOfNode}(jNode)$ 
27    if all route-merging conditions are satisfied then
28       $sol \leftarrow \text{mergeRoutesUsingEdge}(e, iRoute, jRoute, sol)$ 
29    end
30     $\text{deleteEdgeFromList}(e, L)$ 
31  end
32   $sol \leftarrow \text{localSearch}(sol)$ 
33  return  $sol$ 
34 End

```

3.1.3.2 Introducing a Local Search

Once an initial and feasible solution is constructed, a fast local search is applied to improve its quality. This local search mechanism is based on a *2-opt* movement plus a memory-based data structure (hash map) that keeps the best-found route sequence for each set of nodes (Juan et al., 2011). In other words, this mechanism stores the best-known order to travel a specific set of nodes, discovered during the search process, which is frequently updated

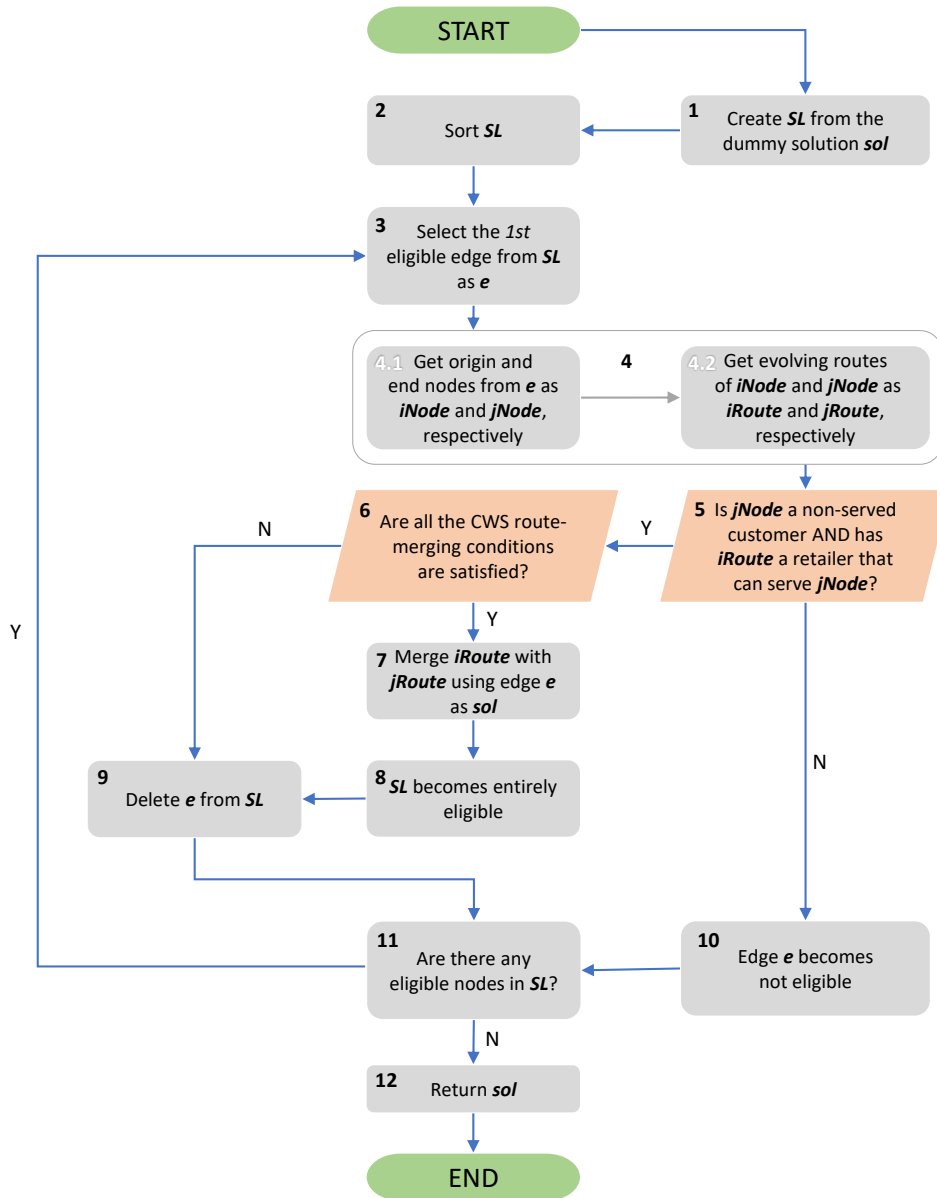
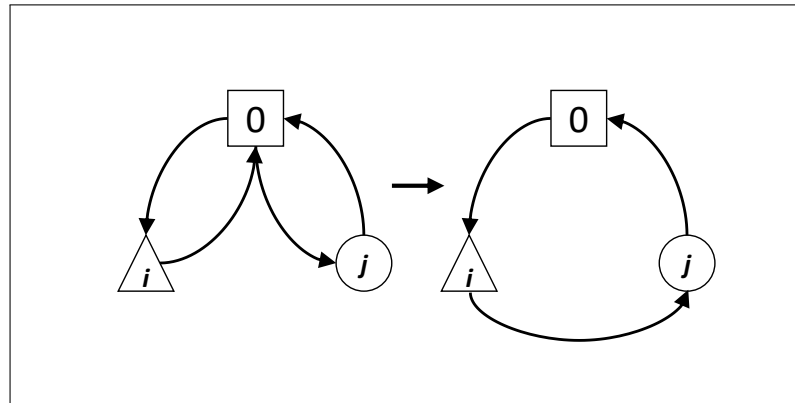


Figure 3.3: The flowchart of CWS₁.

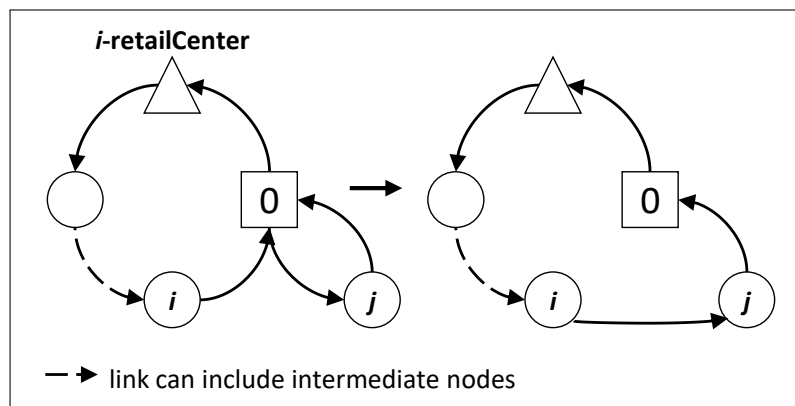
whenever a better cost sequence is found for the same set of nodes. By employing an efficient data structure, such as a hash map or hash table, the accessing and searching for these stored routes is done fastly, then avoiding spending additional computational time when exploring the solution space of determined neighborhood movement.

The *2-opt* is a well-known and popular local search heuristic that has been largely applied to solve traveling salesman and vehicle routing problems. This method is based on removing two edges (links) between nodes and reconnecting them with two new others, being the edges between the removed ones reversed. Figure 3.5 presents the idea on which *2-opt* relies, where red edges (dotted lines) are removed, and replaced by the (new) green edges. The edges that are not removed, but affected by this movement, are reversed.

Since this problem has some precedence particularities, the *2-opt* movement is restricted



(a) Merging between a dummy retail route and a dummy customer route, in which the retail i supplies the customer j .



(b) Merging between a non-dummy retail route and a dummy customer route, in which the retail assigned to customer i supplies the customer j .

Figure 3.4: Possible merging cases in stage 2 (CWS_1).

to movements that do not violate the precedence order between customers and their suppliers –hence, the feasibility of the solution is preserved. Figure 3.6 depicts the 2-opt for the OCVRP. In this case, Figure 3.6a presents the current configuration of a route composed of two retail centers (A and B) and four customers. Each customer is characterized by its corresponding retail center, i.e., the one which supplies the consumer. Accordingly, Figure 3.6b presents a feasible movement, since both retailers are visited before visiting their respective customers, which are visited in a different order as a result of the movement. However, in Figure 3.6c, this movement is not possible due to the visiting of a customer which requires an order from retail B , which has not been visited yet.

3.1.3.3 Extending to a Biased-Randomized Algorithm

To modify the original greedy and deterministic behavior of the LH heuristic, the selection of candidates from the SL is randomized by introducing a skewed probability distribution into the selection process. For the biased-randomized component, we employ the geometric distribution, which is controlled by a single parameter, $\beta \in [0, 1]$. When the β value towards 1, it increases the probability that the highest saving merges are selected. On the contrary,

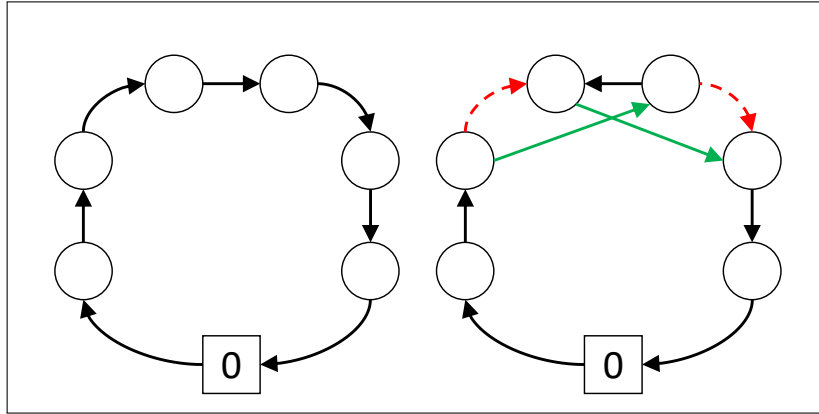


Figure 3.5: The 2-opt movement.

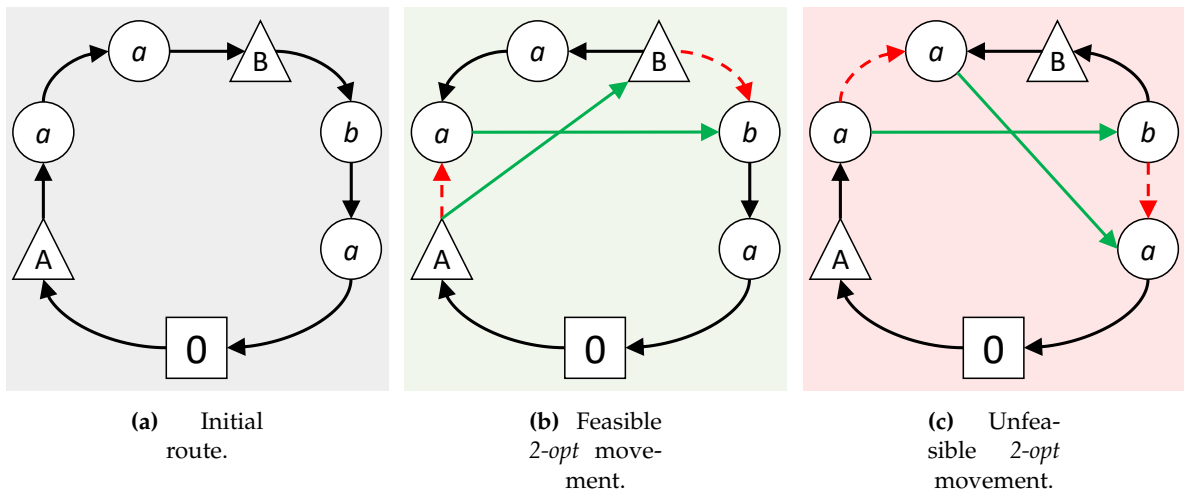


Figure 3.6: The 2-opt movement for the OCVRP.

as β approaches 0, a near-uniform randomization is obtained. In other words, β controls the level at which the selection probabilities decrease along the sorted SL. Hence, unlike deterministic heuristics, which always generate the same solution when starting from the same initial solution, different decisions are taken at each selection iteration, consequently generating different solutions.

Pseudocode 6 represents the selection process of *LH* with biased-randomization, which replaces the greedy selection of edges from the SL. Thus, in its extended version, *BRLH*, the selection of an edge from the sorted list is made accordingly to the probability provided by the parameter β (line 3). Therefore, the greedy behavior is smoothed and the methodology is able to generate different solutions, which allows the exploration of different solution spaces when applying the local search mechanism. In this regard, lines 5 and 21 of Pseudocode 5 are replaced by the method *brSelection* (Pseudocode 6) in order to incorporate the BR strategy into the selection process. Note that, at stage 3 of Pseudocode 5, all the edges of L are eligible, then $l = |L|$ (line 2).

Considering all of the stages which have been introduced, Pseudocode 7 represents the structure of the proposed biased-randomized algorithm with local search (*BRLH*).

Pseudocode 6: Biased-Randomized Selection

Data: savings list SL , geometric distribution parameter β

```

1 Function brSelection( $SL, \beta$ ):
2    $l \leftarrow$  getNumberOfEligibleEdgesFromList( $SL$ )
3   Randomly select position  $x \in \{1, \dots, l\}$  according to distribution  $Geom(\beta)$ 
4    $e \leftarrow$  selectTheXthEligibleEdgeFromList( $x, SL$ )
5   return  $e$ 
6 End

```

Pseudocode 7: Biased-Randomized Savings-Based Heuristic

Data: set of nodes V , geometric distribution parameter β

```

1 Function BRLH( $V, \beta$ ):
2    $sol \leftarrow$  createDummySolution( $V$ )
3    $sol \leftarrow$  CWS1( $sol, \beta$ )
4    $sol \leftarrow$  CWS2( $sol, \beta$ )
5    $sol' \leftarrow$  localSearch( $sol$ )
6   return  $sol'$ 
7 End

```

3.1.3.4 The Multi-Start Approach

Through the use of skewed probability distributions, such as the geometric distribution, the resulting $BRLH$ is able to generate different solutions whenever it is called. Therefore, in order to exploit this intrinsic particularity of biased-randomized algorithms, the $BRLH$ is embedded into a multi-start procedure (MS_{BRLH}) in order to obtain a variety of solutions. The multi-start framework belongs to a family of metaheuristics algorithms (Glover and Kochenberger, 2006), which were introduced in order to provide high-quality solutions using a reasonable amount of computation time and memory. Typically, they require a time-consuming parameter tuning process. However, MS_{BRLH} has been devised to work with a reduced number of parameters, providing an excellent trade-off between simplicity, computational time, and solution quality.

Pseudocode 8 presents the overall structure of the MS_{BRLH} . Initially, in this case, a solution is generated by the deterministic savings-based heuristic, i.e., when $\beta = 1$ (line 1). This solution is set as the best-found solution. While a stop criterion is not met (line 2), which can be defined by a maximum number of iterations or running time, new solutions are generated, with $\beta \in [0, 1)$ (line 3), which diversifies the search for solutions in different regions of the solution space. The best-found solution, which is updated every time a new best solution –that one with a lower cost– is found (line 4), is returned at the end of this process (line 6).

3.1.3.5 The Simheuristic Approach

Deterministic travel times are widely assumed in transportation problems. However, when dealing with real-life problems, which are often fraught with uncertainty, travel times are usually stochastic in nature. As introduced in Section 3.1.2.2, the deterministic travel time employed to traverse edge (i, j) , t_{ij} , can be seen as the travel time required under ideal traffic

Pseudocode 8: Multi-Start Biased-Randomized Savings-Based Heuristic

Data: set of nodes V , geometric distribution parameter β

```

1 Function MSBRLH( $V, \beta$ ):
2    $best_{sol} \leftarrow$  BRLH( $V, 1.0$ )
3   while stop condition is not satisfied do
4      $current_{sol} \leftarrow$  BRLH( $V, \beta$ )
5     updateBestSolution( $best_{sol}, current_{sol}$ )
6   end
7   return  $best_{sol}$ 
8 End

```

conditions. In the proposed extension, the stochastic time to traverse (i, j) is computed as $T_{ij} = t_{ij} + D_{ij}$, where D_{ij} follows a $logNormal(\mu, \sigma)$ probability distribution, and represents a random delay resulting from uncertain conditions. Since the $logNormal$ probability distribution can only take positive values, it follows that $T_{ij} > t_{ij}, \forall (i, j)$ in the set of connecting edges.

Pseudocode 9 describes the proposed simheuristic approach, which is henceforth named as SIM_{BRLH} . Initially, a solution is generated by the savings-based $BRLH$ heuristic, introduced in Section 3.1.3.3, in line 1 (Pseudocode 7), by employing the greedy approach (i.e., $\beta \approx 1$). A short MCS is then performed on this initial solution (line 2), in order to estimate its average stochastic cost. Each simulation run replaces the deterministic travel times of a solution with randomly sampled ones –according to the assumed probability distribution. This initial solution is set as the best-found stochastic solution cost (line 4). While the termination criterion is not met (line 6), different solutions are generated by the $BRLH$ (line 7). The deterministic cost of the initial solution is considered for guiding the search. Therefore, a solution is accepted for being submitted to the simulation module (line 10) only if its deterministic cost is smaller than the best-found deterministic solution cost plus $m\%$ of its value (line 9). This solution filtering approach reduces the amount of time spent on testing unpromising solutions in computationally expensive simulations. Moreover, by allowing the acceptance of moderately worse solutions, controlled by the parameter m , a better exploration of the solution space can be achieved (Talbi, 2009). At this stage, q_{short} MCS runs are used to test the accepted solution, by replacing the deterministic travel times with randomly sampled ones. From this complete simulation process, the average stochastic cost of each solution is computed. Every time a new best stochastic cost is found (line 15), this solution is introduced into a pool of ‘elite’ solutions E (line 17). This process is repeated while the termination criterion is not met. On this reduced set of solutions, q_{long} MCS runs are performed (line 23) in order to generate more accurate results for solutions in stochastic environments. During the simulation process we also obtain an estimate of the reliability rate of a solution (Faulin et al., 2008b). This estimate is computed as the rate at which all routes show completion times lower than the maximum allowed travel time. At the end, the set of elite solutions is sorted in descending order of their expected cost (line 25), and the best-found stochastic solution is provided to the decision-maker.

By developing this simheuristic approach for solving the SOCVRP, apart from holding

Pseudocode 9: Simheuristic Approach

Data: set of nodes V , geometric distribution parameter β , acceptance margin m , number of short simulations q_{short} , number of long simulations q_{long} , log normal distribution parameters μ and σ , maximum number of iterations max_{iter}

```

1 Function SIMBRLH( $V, \beta, m, q_{short}, q_{long}, \mu, \sigma, max_{iter}$ ):
2    $baseSol \leftarrow$  BRLH( $V, 1.0$ )
3   simulation( $baseSol, q_{short}, \mu, \sigma$ )
4    $bestCost_d \leftarrow baseSol.getDeterministicCost()$ 
5    $bestCost_s \leftarrow baseSol.getStochasticCost()$ 
6    $n_{iter} \leftarrow 0$ 
7   while  $n_{iter} < max_{iter}$  do
8      $sol \leftarrow$  BRLH( $V, \beta$ )
9      $cost_d \leftarrow sol.getDeterministicCost()$ 
10    if  $cost_d < bestCost_d + bestCost_d \times m$  then
11      simulation( $sol, q_{short}, \mu, \sigma$ )
12       $cost_s \leftarrow sol.getStochasticCost()$ 
13      if  $cost_d < bestCost_d$  then
14         $bestCost_d \leftarrow cost_d$ 
15      end
16      if  $cost_s < bestCost_s$  then
17         $bestCost_s \leftarrow cost_s$ 
18         $E \leftarrow E \cup \{sol\}$ 
19      end
20    end
21     $n_{iter} \leftarrow n_{iter} + 1$ 
22  end
23  foreach  $sol \in E$  do
24    simulation( $sol, q_{long}, \mu, \sigma$ )
25  end
26   $E \leftarrow$  sort( $E$ )
27   $bestStochSol \leftarrow E.get(0)$ 
28  return  $bestStochSol$ 
29 End

```

the possibility of optimizing the system's performance, this strategy is also able to measure the probability of deploying a plan without any route failure in a realistic scenario under uncertain travel times.

3.1.4 Computational Experiments and Results

The proposed methodologies were tested on the benchmark of 60 large-sized instances proposed by Abdulkader et al. (2018). These instances are different in the number of retail stores and customers. The first 20 instances are considered small-sized and were optimally solved with CPLEX by the authors. The remaining 60 ones are large-sized instances which are different in the inventory scenarios of the retail centers (tight, relaxed and abundant). According to Abdulkader et al. (2018), for each scenario, the total network inventory is computed as:

- Tight:

$$\sum_{i \in R} I_{ip} = \sum_{j \in C} D_{jp} + U[0.1, 0.2] \sum_{j \in C} D_{jp} \quad \forall p \in P \quad (3.5)$$

- Relaxed:

$$\sum_{i \in R} I_{ip} = \sum_{j \in C} D_{jp} + U[0.5, 1.0] \sum_{j \in C} D_{jp} \quad \forall p \in P \quad (3.6)$$

- Abundant: (at each retail store)

$$I_{ip} = \sum_{j \in C} D_{jp} \quad \forall i \in R, \forall p \in P \quad (3.7)$$

where P is the set of heterogeneous products, R is the set of retail centers, C is the set of customers, $\sum I_{ip}$ is the total inventory available of a product p , and $\sum D_{jp}$ is the total online demand of a product p .

The maximum tour length and the vehicle capacity are fixed to 8 hours and 100 weight units, respectively. Regarding the BR process during the solution-construction stage, after preliminary performance tests, the parameter β is randomly selected within the interval $[0.45, 0.75]$. Regarding the maximum run time $time_{max}$, the value was set depending on the size of the instance $(c + r) \times 0.342$, which leads to a maximum execution time of 60 seconds in the case of the largest instance. The same parameters set is considered for both the deterministic and stochastic problem variants. The entire algorithm was coded in Java, and the tests were performed on an Intel Core i7-8550U processor with 16 GB of RAM.

3.1.4.1 Results for the Deterministic Omnichannel Vehicle Routing Problem

The initial analysis aims to compare the solutions generated by the greedy heuristic (LH) and by the MS_{BRLH} –in which β is (uniformly) randomly selected in the interval $[0.45, 0.75]$ – with the solutions obtained by the two-phase heuristic (AH) and multi-ant colony metaheuristic (MAC) proposed by Abdulkader et al. (2018). Their methodologies were performed on four 2.1 GHz processors with 16-cores each and a total of 256 GB RAM. In the case of MS_{BRLH} , 10 runs were performed for each instance. The performance of the heuristics was measured by means of the percentage gap between the best-found solution using that methodology, i.e., the one with lowest cost value, and the best-found solution obtained with the alternative solution methodology. Thus, the lower the gap is, the better the performance of the method is. The solution cost is given in travel time (in minutes). While negative gaps between methods A and B ($A-B$) correspond to worse solution costs obtained by solving methodology A , positive gaps correspond to better solutions obtained by method A .

Tables 3.1, 3.2, and 3.3 present the results obtained for tight, relaxed, and abundant inventory scenarios, respectively. For each problem instance (I), we present results for: the cost of the best-found solution obtained by the different methodologies; the average cost of the MS_{BRLH} ; the CPU time (in seconds) required by each methodology; and their percentage gaps. The best results returned by the solution methodologies are highlighted in bold.

By analyzing Tables 3.1, 3.2, and 3.3, we can observe that the MS_{BRLH} algorithm is able to improve previous results (from LH) by 9-12%, on average, presented in column *gap* (2-1). It is an expected behavior since the multi-start approach, unlike the deterministic heuristic, relies on several executions –according to the stop criterion– of the biased-randomized variant of

Table 3.1: Comparison of the results obtained by the LH and MS_{BRLH} solution methods with those obtained by Abdulkader et al. (2018)'s methods (AH and MAC) in the tight inventory scenario.

I	$ R $	$ C $					Avg. Cost (2)	% SD (2)	Time (sec.)				gap			
			1 LH	2 MS_{BRLH}	3 AH	4 MAC			(1)	(2)	(3)	(4)	(1-2)	(1-3)	(3-2)	(4-2)
b1	10	25	1,277.5	1,110.9	1,631.6	1,002.5	1,119.6	0.9%	0	7	0	7	-13%	-22%	-32%	11%
b2	10	50	1,641.1	1,378.2	2,057.5	1,192.0	1,392.0	1.4%	0	8	0	47	-16%	-20%	-33%	16%
b3	10	75	2,663.7	2,437.2	3,006.2	1,815.4	2,450.7	2.3%	0	25	0	79	-9%	-11%	-19%	34%
b4	10	100	2,415.1	1,930.3	2,830.2	1,529.0	1,980.7	1.9%	0	15	0	286	-20%	-15%	-32%	26%
b5	10	150	2,678.2	2,395.3	3,478.7	1,905.2	2,408.9	1.1%	0	38	0	576	-11%	-23%	-31%	26%
b6	15	25	1,540.5	1,389.4	1,774.4	1,313.7	1,400.6	0.8%	0	4	0	7	-10%	-13%	-22%	6%
b7	15	50	2,059.0	1,769.3	2,461.8	1,522.3	1,803.7	2.1%	0	0	0	44	-14%	-16%	-28%	16%
b8	15	75	3,105.3	2,620.5	3,545.1	2,101.8	2,630.3	0.5%	0	6	0	131	-16%	-12%	-26%	25%
b9	15	100	3,121.5	2,836.9	3,529.0	2,329.5	2,860.5	0.7%	0	1	0	209	-9%	-12%	-20%	22%
b10	15	150	4,292.4	3,787.2	4,916.8	3,012.2	3,797.4	0.8%	0	45	0	430	-12%	-13%	-23%	26%
b11	20	25	2,035.2	1,817.1	2,432.6	1,611.3	1,838.4	1.0%	0	10	0	11	-11%	-16%	-25%	13%
b12	20	50	2,335.4	2,109.1	2,695.3	1,800.9	2,112.8	0.3%	0	1	0	50	-10%	-13%	-22%	17%
b13	20	75	3,212.7	2,765.1	3,936.7	2,406.0	2,796.2	1.4%	0	12	0	127	-14%	-18%	-30%	15%
b14	20	100	3,025.2	2,842.7	3,826.1	2,483.8	2,881.4	0.7%	0	20	0	327	-6%	-21%	-26%	14%
b15	20	150	3,934.3	3,308.0	4,496.1	2,679.2	3,332.1	1.4%	0	50	0	708	-16%	-12%	-26%	23%
b16	25	25	2,019.4	1,847.2	2,254.9	1,669.6	1,858.0	0.7%	0	8	0	13	-9%	-10%	-18%	11%
b17	25	50	2,665.9	2,434.8	3,020.8	1,965.6	2,442.8	0.6%	0	17	0	46	-9%	-12%	-19%	24%
b18	25	75	3,207.6	2,853.4	3,963.5	2,449.8	2,885.8	1.1%	0	18	0	136	-11%	-19%	-28%	16%
b19	25	100	4,064.0	3,551.0	4,933.9	2,788.5	3,588.7	1.2%	0	42	0	257	-13%	-18%	-28%	27%
b20	25	150	3,782.7	3,512.8	4,721.3	2,890.3	3,525.5	1.2%	0	17	0	712	-7%	-20%	-26%	22%
<i>Average</i>								1.1%	0	17	0	210	-12%	-16%	-26%	19%

Table 3.2: Comparison of the results obtained by the LH and MS_{BRLH} solution methods with those obtained by Abdulkader et al. (2018)'s methods (AH and MAC) in the relaxed inventory scenario.

I	$ R $	$ C $					Avg. Cost (2)	% SD (2)	Time (sec.)				gap			
			1 LH	2 MS_{BRLH}	3 AH	4 MAC			(1)	(2)	(3)	(4)	(1-2)	(1-3)	(3-2)	(4-2)
b21	10	25	1,233.0	1,030.0	1,571.6	879.2	1,048.9	1.5%	0	5	0	10	-16%	-22%	-34%	17%
b22	10	50	1,490.8	1,275.7	1,920.6	1,083.7	1,293.9	1.4%	0	4	0	85	-14%	-22%	-34%	18%
b23	10	75	2,468.0	2,021.9	2,699.2	1,591.5	2,049.5	0.7%	0	18	0	167	-18%	-9%	-25%	27%
b24	10	100	1,885.0	1,632.2	2,305.1	1,437.7	1,645.3	1.4%	0	7	0	528	-13%	-18%	-29%	14%
b25	10	150	1,998.6	1,980.4	2,700.4	1,520.5	1,981.8	0.8%	0	39	0	1,836	-1%	-26%	-27%	30%
b26	15	25	1,591.4	1,268.0	1,665.2	1,180.8	1,308.9	0.6%	0	4	0	11	-20%	-4%	-24%	7%
b27	15	50	1,940.7	1,652.1	2,320.7	1,329.3	1,660.9	0.7%	0	9	0	73	-15%	-16%	-29%	24%
b28	15	75	2,436.3	2,101.7	3,016.5	1,692.4	2,160.4	0.5%	0	3	0	279	-14%	-19%	-30%	24%
b29	15	100	2,648.3	2,395.4	3,302.4	2,016.4	2,412.5	0.9%	0	25	0	567	-10%	-20%	-27%	19%
b30	15	150	3,373.2	2,819.0	3,919.0	2,399.6	2,847.6	1.1%	0	18	0	1,407	-16%	-14%	-28%	17%
b31	20	25	1,835.5	1,679.1	1,993.5	1,495.8	1,682.7	0.4%	0	11	0	16	-9%	-8%	-16%	12%
b32	20	50	2,320.5	1,960.7	2,713.0	1,656.9	1,965.7	0.6%	0	6	0	76	-16%	-14%	-28%	18%
b33	20	75	2,404.6	2,267.2	3,393.3	1,799.6	2,269.8	0.9%	0	28	0	262	-6%	-29%	-33%	26%
b34	20	100	2,751.9	2,469.3	3,127.5	2,018.5	2,490.6	1.0%	0	34	0	740	-10%	-12%	-21%	22%
b35	20	150	3,157.4	2,818.0	3,742.2	2,291.0	2,824.5	1.1%	0	32	0	2,141	-11%	-16%	-25%	23%
b36	25	25	1,844.0	1,683.7	2,032.1	1,550.0	1,700.9	1.1%	0	13	0	15	-9%	-9%	-17%	9%
b37	25	50	2,663.9	2,322.3	3,130.5	1,939.5	2,357.5	1.1%	0	22	0	73	-13%	-15%	-26%	20%
b38	25	75	2,790.7	2,559.7	3,433.2	2,088.6	2,569.7	0.9%	0	30	0	283	-8%	-19%	-25%	23%
b39	25	100	3,352.9	3,038.1	3,824.5	2,244.1	3,053.6	0.9%	0	27	0	656	-9%	-12%	-21%	35%
b40	25	150	2,971.1	2,830.6	3,447.9	2,229.4	2,837.7	0.6%	0	37	0	2,077	-5%	-14%	-18%	27%
<i>Average</i>								0.9%	0	19	0	565	-12%	-16%	-26%	21%

the heuristic. Therefore, the chances of finding better solutions are higher. Comparing the results obtained by LH with those produced by the two-phase heuristic, column gap (3-1), our results are between 12% and 16% better even without considering any stochasticity in the search-guidance process. When comparing the MS_{BRLH} with the AH heuristic, column

Table 3.3: Comparison of the results obtained by the LH and MS_{BRLH} solution methods with those obtained by Abdulkader et al. (2018)'s methods (AH and MAC) in the abundant inventory scenario.

I	$ R $	$ C $					Avg. Cost (2)	% SD (2)	Time (sec.)				gap			
			1 LH	2 MS_{BRLH}	3 AH	4 MAC			(1)	(2)	(3)	(4)	(1-2)	(1-3)	(3-2)	(4-2)
b41	10	25	805.4	760.5	897.6	711.3	760.5	0.1%	0	1	0	16	-6%	-10%	-15%	7%
b42	10	50	1,014.2	870.4	1,287.8	875.2	871.2	0.0%	0	8	0	143	-14%	-21%	-32%	-1%
b43	10	75	1,463.5	1,259.6	1,531.1	1,132.1	1,266.4	0.7%	0	9	0	358	-14%	-4%	-18%	11%
b44	10	100	1,379.4	1,284.9	1,636.5	1,224.1	1,294.1	0.4%	0	30	0	978	-7%	-16%	-21%	5%
b45	10	150	1,499.3	1,364.3	1,551.8	1,273.9	1,385.3	1.3%	0	43	0	2,085	-9%	-3%	-12%	7%
b46	15	25	1,137.5	1,024.3	1,264.3	996.9	1,028.8	0.4%	0	6	0	22	-10%	-10%	-19%	3%
b47	15	50	1,247.6	1,135.1	1,488.1	1,080.3	1,141.0	0.7%	0	17	0	159	-9%	-16%	-24%	5%
b48	15	75	1,595.3	1,355.2	1,815.2	1,252.4	1,361.2	0.4%	0	17	0	559	-15%	-12%	-25%	8%
b49	15	100	2,021.3	1,777.8	2,242.4	1,594.0	1,798.9	1.2%	0	24	0	1,167	-12%	-10%	-21%	12%
b50	15	150	2,059.6	1,869.2	2,459.5	1,691.4	1,873.6	0.6%	0	26	0	4,126	-9%	-16%	-24%	11%
b51	20	25	1,507.2	1,414.7	1,660.9	1,302.9	1,418.2	0.4%	0	11	0	33	-6%	-9%	-15%	9%
b52	20	50	1,464.7	1,366.8	1,740.7	1,301.0	1,368.4	0.5%	0	7	0	156	-7%	-16%	-21%	5%
b53	20	75	1,797.7	1,591.5	2,096.8	1,421.8	1,599.7	0.6%	0	18	0	605	-11%	-14%	-24%	12%
b54	20	100	2,066.2	1,881.8	2,226.4	1,640.6	1,883.9	2.3%	0	23	0	1,370	-9%	-7%	-15%	15%
b55	20	150	2,214.0	2,025.1	2,518.2	1,763.3	2,030.0	0.4%	0	52	0	5,321	-9%	-12%	-20%	15%
b56	25	25	1,423.2	1,368.1	1,550.7	1,311.6	1,372.3	0.5%	0	7	0	36	-4%	-8%	-12%	4%
b57	25	50	1,670.8	1,559.7	1,835.4	1,468.1	1,570.2	0.8%	0	11	0	203	-7%	-9%	-15%	6%
b58	25	75	2,047.5	1,845.3	2,276.9	1,654.9	1,847.5	1.0%	0	7	0	791	-10%	-10%	-19%	12%
b59	25	100	1,856.4	1,797.8	2,061.9	1,575.7	1,801.0	1.0%	0	15	0	1,262	-3%	-10%	-13%	14%
b60	25	150	1,968.1	1,837.3	2,347.8	1,653.3	1,849.1	0.6%	0	60	0	4,549	-7%	-16%	-22%	11%
<i>Average</i>								0.7%	0	19	0	1,197	-9%	-12%	-19%	9%

gap (3-2), MS_{BRLH} approach is able to reduce solution costs even more, by up to 26% in short computational times (about 18 seconds on average). On the other hand, when comparing the MS_{BRLH} with MAC approach, column gap (4-2), its solutions are between 9% and 21% worse, on average. Particularly, in the abundant inventory scenario, MS_{BRLH} 's results are only 9% worse than MAC . Notice, however, that the processing time required by MAC is substantially larger in all inventory scenarios.

Figures 3.7 and 3.8 present how both the gap and the cost of the solutions, i.e., the objective function (OF) value, behave according to the employed solution approach and inventory scenario, respectively.

By analyzing Figure 3.7, which addresses the performance of our multi-start approach and MAC metaheuristic for the three different considered inventory scenarios, in terms of gap, it is evident that MS_{BRLH} performs better in the abundant inventory scenario, being able to find one better solution and several others with a maximum gap of 8%. Moreover, we can observe a variability of around 10% in the gap between MS_{BRLH} and MAC , on average, for the majority of instances (between the first and third quartiles). This variability is reduced to around 8% for the relaxed and abundant scenarios. These results demonstrate the robustness of the MS_{BRLH} solution approach for the deterministic case. When analyzing Figure 3.8, which presents the overall performance of each solution approach for each inventory scenario, we can observe that the multi-start strategy is more efficient than both LH and AH heuristics, by generating solutions with a lower cost. To complement these box-plots, an ANOVA test was run for each inventory scenario. The p -values associated with the tight, relaxed, and abundant inventory scenarios were, respectively, of 0.001, 0.000, and 0.000. Also, Fisher's Least Significant Difference test suggests significant differences in all

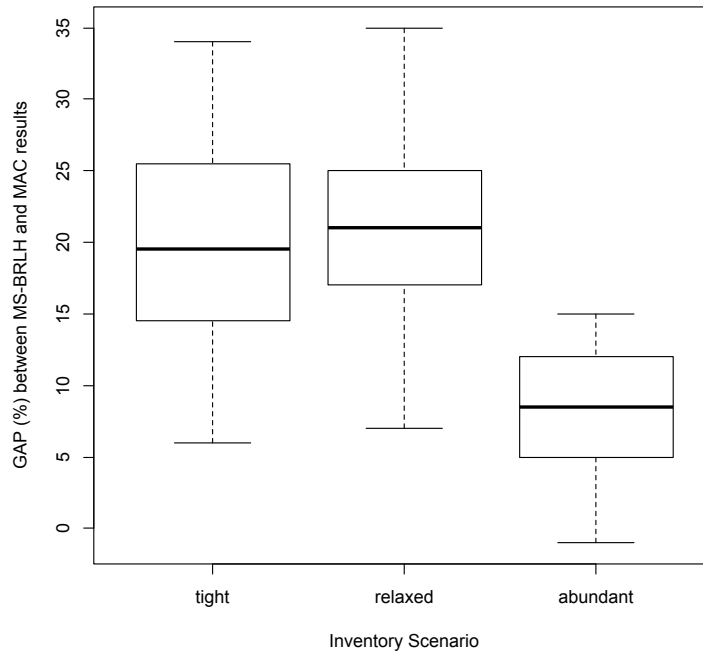


Figure 3.7: Gap between the best-found solutions (from MS_{BRLH}) and the MAC's results, for each inventory scenario.

three scenarios between MAC and AH , between MS_{BRLH} and AH , as well as between MAC and LH . However, cost differences between MAC and MS_{BRLH} were not statistically significant in any scenario, although MAC employed a noticeably higher amount of computing time than MS_{BRLH} .

Next, in Figure 3.9, we present the convergence of three problem instances' solutions, including one from each different inventory scenario (instances $b6$, $b26$, and $b46$), by comparing the MS_{BRLH} with the best-known solutions, in terms of gap. As introduced, these instances require approximately 15 seconds of processing time, given their magnitude.

As we can see in Figure 3.9, the instances demonstrate the same convergence behavior as solution time increases. It is noticeable that the convergence rate is abrupt during the first few seconds. However, contrary to the tight and relaxed inventory scenarios, where the solutions continue improving over time, the search demonstrates more stable convergence during the remaining execution time in the abundant scenario. Being more efficient in the more flexible inventory case, the multi-start approach quickly achieves its best solutions.

3.1.4.2 Results for the Stochastic Omnichannel Vehicle Routing Problem

While the $BRLH$ is guided by a single parameter, β , the simheuristic approach is also controlled by the maximum running time $time_{max}$, the acceptance margin of worst solutions m , and the number, q_{short} and q_{long} , of simulation repeats in short and long simulation runs, respectively. The stochastic travel times for each edge are set by the log-normal distribution parameters, μ and σ .

Table 3.4 summarizes the setup of the parameters employed during the computational experiments. For calibrating these parameters, we have used the methodology proposed in

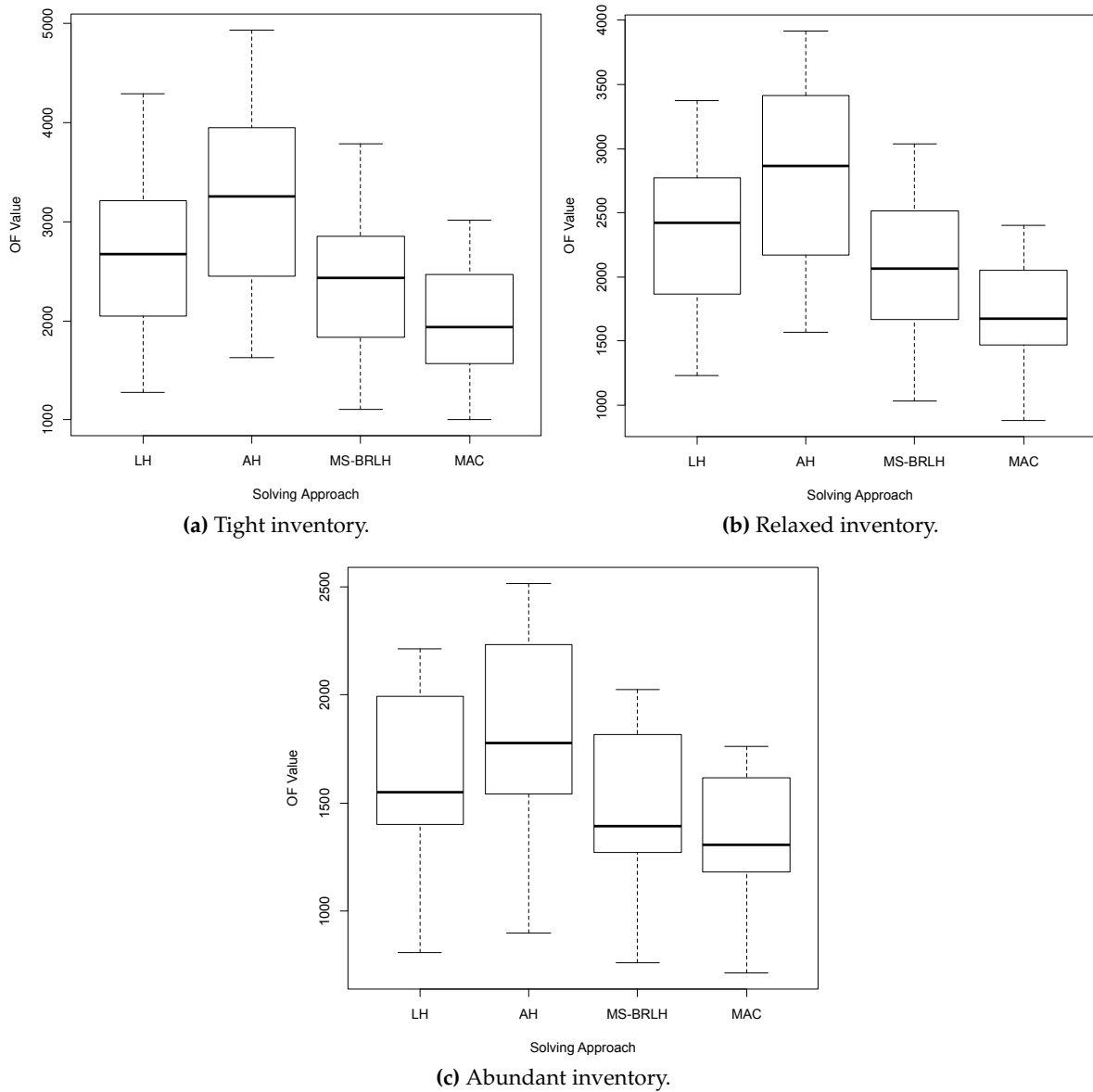


Figure 3.8: Comparison of the solutions cost (OF value) from each solving approach, for each inventory scenario.

Calvet et al. (2016b), which is based on a general and automated statistical learning procedure. Notice that three different values have been considered for the σ parameter, which is used to modify the deterministic travel times. Since the maximum tour length is fixed independently of the size of the instances, small-sized instances are more likely to generate short routes. Therefore, a larger value for σ introduces more variability in the travel times, which increases route failure rates. On the other hand, large-sized instances are often composed of larger routes, then a small value for σ is introduced. In particular we have, $\sigma = 2.5$ for small instances (composed of 25 customers), $\sigma = 1.55$ for large instances (composed of 150 customers), and $\sigma = 1.9$ for the remaining medium-sized ones. This approach ensures that small, medium, and large instances each have similar levels of difficulty with respect to the risk of route failure.

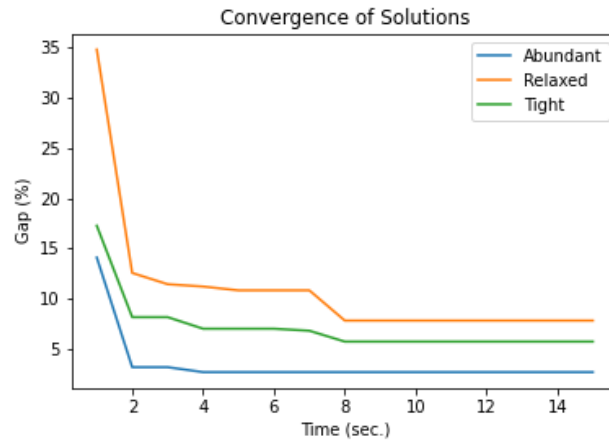


Figure 3.9: Comparison between MS_{BRLH} and MAC on solving large-sized instances.

Table 3.4: Parameter setup.

β	m	μ	σ	q_{short}	q_{long}	$time_{max}$
[0.45, 0.75]	20%	0	{1.55, 1.9, 2.5}	100	1000	$(r + c) \times 0.342$

Considering that the deterministic OCVRP and related solution approaches have been analyzed in Section 3.1.4.1, the current analysis aims to measure the quality of the solutions generated for the deterministic scenario (MS_{BRLH}) against the solutions generated for the stochastic scenario (SIM_{BRLH}). Since the only difference between the deterministic and stochastic scenarios is that stochastic delays are added to edge traversal times, we can consider the deterministic cost of the best solutions generated by MS_{BRLH} as a lower bound (LB) of the stochastic travel times of the best SIM_{BRLH} solution. Moreover, since MS_{BRLH} does not account for stochastic travel times, we can consider the stochastic travel time of the best MS_{BRLH} solution as an upper bound (UB) for the stochastic travel time of the best SIM_{BRLH} solutions. Table 3.5 provides both LBs and UBs values and the best-found stochastic travel times obtained by the SIM_{BRLH} . The solutions reported in the SIM_{BRLH} column are the best-found stochastic travel times.

As we can see in Table 3.5, all the SIM_{BRLH} solution costs are between the LB and the UB, as expected. For 24 problem instances, the solution returned by the simheuristic is better than the best deterministic solution when it is tested in the stochastic scenario (the UB column). From this, we can assert that the SIM_{BRLH} is able to generate competitive results for the stochastic scenario. The reliability value is calculated by simulation for each solution and represents the probability that all routes are completed within maximum tour duration. For visualizing this trade-off between the deterministic cost of the solutions and their reliability rate, which are conflicting objectives, a Pareto frontier of non-dominated is presented. Accordingly, Figures 3.10a, 3.10b, and 3.10c present the non-dominated solutions for three different instances ($b17$, $b37$, and $b57$), each one belonging to a different inventory scenario. A solution is non-dominated if, no other solution has a greater reliability and a lower or equal travel cost, or if no other solution has a lower travel cost and a greater or equal

Table 3.5: Analysis of the results obtained by the SIM_{BRLH} on scenarios of tight, relaxed and abundant inventory.

<i>I</i>	<i>Tight inventory</i>			<i>I</i>	<i>Relaxed inventory</i>			<i>I</i>	<i>Abundant inventory</i>		
	<i>LB</i>	SIM_{BRLH}	<i>UB</i>		<i>LB</i>	SIM_{BRLH}	<i>UB</i>		<i>LB</i>	SIM_{BRLH}	<i>UB</i>
b1	1,110.9	2,164.5	2,167.0	b21	1,030.0	2,096.3	2,154.6	b41	760.5	1,731.0	1,858.7
b2	1,378.2	2,084.7	2,084.7	b22	1,275.7	1,996.6	1,996.6	b42	870.4	1,574.1	1,636.8
b3	2,437.2	3,465.3	3,465.3	b23	2,021.9	3,043.9	3,043.9	b43	1,259.6	2,239.4	2,239.4
b4	1,930.3	3,184.3	3,184.3	b24	1,632.2	2,891.1	2,989.5	b44	1,284.9	2,553.4	2,553.4
b5	2,395.3	3,790.2	3,790.2	b25	1,980.4	3,363.6	3,363.6	b45	1,364.3	2,753.9	2,753.9
b6	1,389.4	2,628.3	2,775.6	b26	1,268.0	2,487.9	2,487.9	b46	1,024.3	2,269.2	2,269.2
b7	1,769.3	2,544.9	2,544.9	b27	1,652.1	2,446.5	2,446.5	b47	1,135.1	1,919.3	1,944.4
b8	2,620.5	3,715.6	3,715.6	b28	2,101.7	3,178.6	3,178.6	b48	1,355.2	2,420.6	2,442.4
b9	2,836.9	4,209.9	4,209.9	b29	2,395.4	3,725.6	3,725.6	b49	1,777.8	3,138.7	3,146.9
b10	3,787.2	5,258.7	5,258.7	b30	2,819.0	4,269.7	4,269.7	b50	1,869.2	3,308.8	3,312.6
b11	1,817.1	3,261.5	3,401.5	b31	1,679.1	3,099.3	3,099.3	b51	1,414.7	2,814.0	3,095.4
b12	2,109.1	2,978.8	2,983.5	b32	1,960.7	2,819.4	2,832.9	b52	1,366.8	2,233.4	2,250.7
b13	2,765.1	3,917.1	3,917.1	b33	2,267.2	3,412.4	3,412.4	b53	1,591.5	2,732.9	2,732.9
b14	2,842.7	4,312.3	4,312.3	b34	2,469.3	3,891.5	3,914.8	b54	1,881.8	3,289.1	3,304.1
b15	3,308.0	4,813.7	4,813.7	b35	2,818.0	4,311.4	4,311.4	b55	2,025.1	3,510.5	3,510.5
b16	1,847.2	3,460.4	3,591.2	b36	1,683.7	3,267.0	3,273.6	b56	1,368.1	2,944.6	3,180.0
b17	2,434.8	3,377.2	3,393.7	b37	2,322.3	3,266.1	3,268.6	b57	1,559.7	2,497.4	2,497.4
b18	2,853.4	4,082.6	4,082.6	b38	2,559.7	3,782.0	3,788.0	b58	1,845.3	3,065.6	3,065.6
b19	3,551.0	5,039.2	5,039.2	b39	3,038.1	4,548.9	4,548.9	b59	1,797.8	3,287.3	3,287.3
b20	3,512.8	5,088.2	5,088.2	b40	2,830.6	4,384.5	4,385.9	b60	1,837.3	3,380.0	3,380.0

reliability level. The $b17$ solution was randomly chosen, while the $b37$ and $b57$ solutions are for the same problem but set in the two other inventory scenarios. The square orange dot represents the best deterministic solution found by the MS_{BRLH} (Section 3.1.3.4), while the remaining ones, round and blue, represent different solutions with a higher reliability rate, but with higher operating costs.

As we can see in Figure 3.10, despite being the solutions with the lowest cost, the best deterministic solutions (square dots) are the least reliable ones for stochastic scenarios. Particularly in Figure 3.10a, the best deterministic solution is approximately only 21% reliable under stochasticity. By selecting higher-cost solutions, the reliability rate reaches more than 70%. The same behavior is noticed in Figures 3.10b and 3.10c, however, the deterministic $b57$ solution is reasonably reliable. Usually, low-cost solutions are made up of a small number of large vehicle routes, in terms of travel distance and time. Therefore, when increasing the travel time variability in those scenarios, the risk of exceeding the time constraint is higher. On the other hand, higher cost solutions are built from a larger number of smaller routes, and smaller routes exhibit a lower risk of violating the maximum tour duration constraint in stochastic scenarios. In this way, decision-makers should consider that low-cost solutions under deterministic scenarios might not necessarily be the best option when stochasticity is taken into account.

3.1.5 Conclusions

With the emergence of online retail channels and the popularization of mobile devices, new retailing modes have become popular. Some of these retailing practices allow customers to browse through different online stores and, then, to get the items bought directly delivered to their homes. Hence, new versions of the vehicle routing problem (VRP) considering

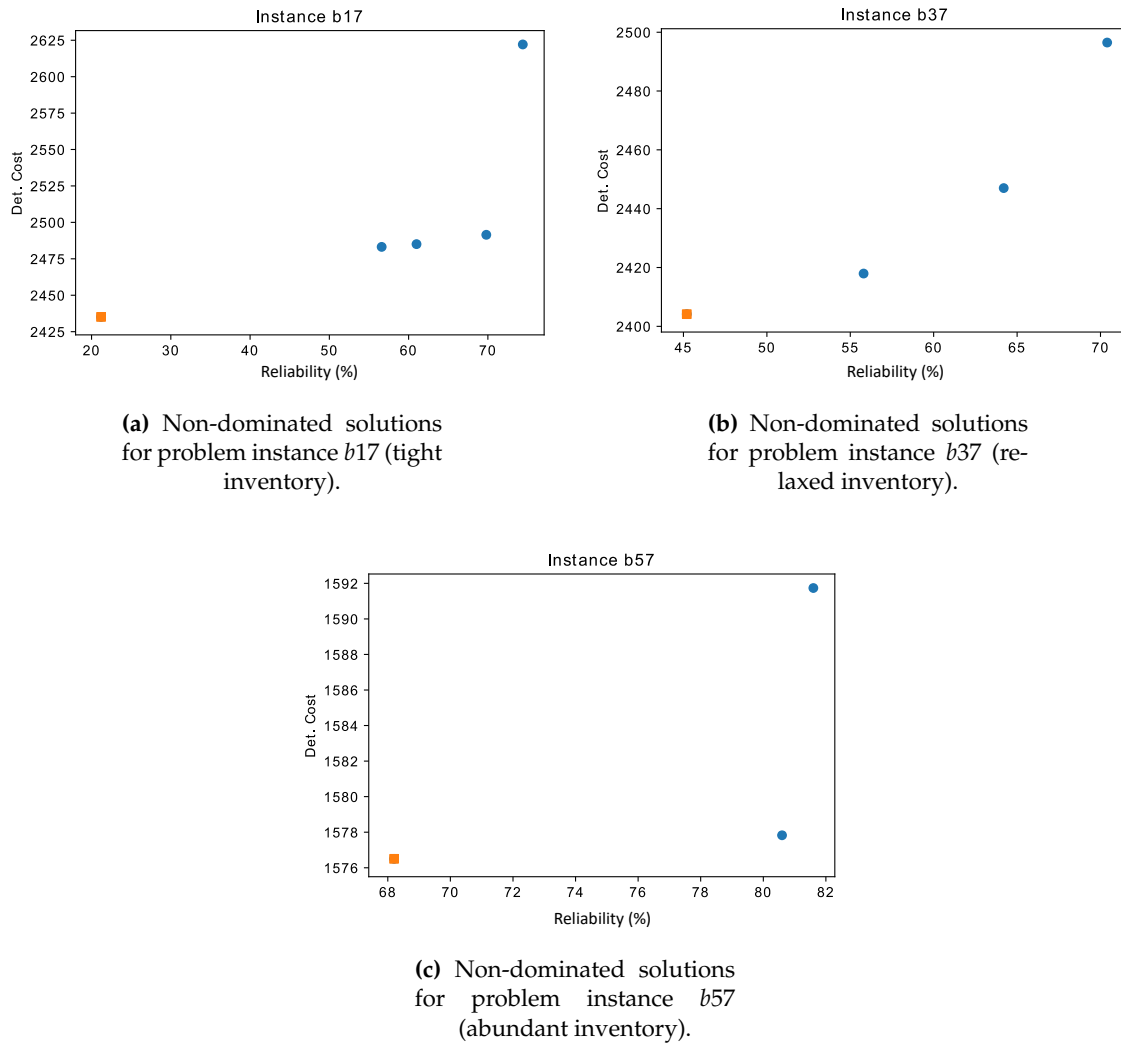


Figure 3.10: Set of non-dominated solutions of problem instances *b17* (a), *b37* (b) and *b57* (c).

additional decision variables and constraints have emerged. Omnichannel retailing leads to an integrated problem combining the VRP and the pick-up and delivery problem. In omnichannel distribution systems, a set of retail stores need to be replenished and, at the same time, products have to be sent from these stores to final customers. The resulting omnichannel VRP consists in two stages: (i) a group of retail stores that must be served from a distribution center; and (ii) a set of online consumers who must be served, by the same fleet of cargo vehicles, from these retail stores.

In this chapter, two versions of the OCVRP were addressed. On contrary to the deterministic variant, in which the travel times are given and fixed, a more realistic version of the deterministic OCVRP, where travel times are modeled as random variables following a log-normal distribution, is presented for the first time in the literature. For solving the deterministic OCVRP, a simple heuristic was initially introduced. This heuristic was then

extended into a multi-start biased-randomized algorithm, which is tested against the state-of-the-art methodologies. The multi-start biased-randomized algorithm has performed reasonably well in a set of 60 instances of the deterministic OCVRP, being able to generate reasonably good solutions in few seconds, of orders of magnitude smaller than the alternative solution methods from the literature, which require up to 4,549 seconds in the extreme case, for generating a solution only 11% less costly than ours. To cope with stochastic OCVRP, a simheuristic approach was proposed, due to the satisfactory efficiency of the solution methods to solve the OCVRP. Apart from reducing the operating costs of this system, the proposed simheuristic approach is also capable of measuring the reliability of any proposed solution when it is employed in a stochastic scenario. To the best of our knowledge, this is the first time that such a stochastic variant of the problem has been solved in the literature. Regarding the simheuristic results, we conclude that the best deterministic solutions may perform badly when used in a stochastic scenario. Those solutions are often not reliable in terms of completing all routes within a time limit. On the other hand, the simheuristic approach was able to generate reliable and competitive results for these stochastic scenarios. Therefore, the proposed methodology enables decision-makers to choose the solution that better fits his or her utility function in terms of cost and reliability level.

3.2 The Vehicle Routing Problem with Optional Backhauls

Reverse logistics and closed-loop supply chains have been increasingly studied in recent years. Both concepts are related to the return of products or materials from the point of consumption in order to recover value. In particular, Govindan and Soleimani (2017) found that most addressed issues are re-manufacturing and waste management, while the topic of package recovery is barely tackled despite its environmental impact (Kroon and Vrijens, 1995). Given these considerations, disposable packages have been replaced by returnable transport items (RTIs), e.g., reusable pallets, trays, boxes, or any other mean to assemble goods (ISO, 2016). Still, environmental issues are not the only concern regarding RTIs management. According to Glock (2017), RTIs are an important asset for many industries, since they can decrease the selling cost for customers.

Examples from different industries highlight the importance of RTIs in real-world transportation practices. For instance, in agri-food supply chains, it is usual that products are harvested and transported in boxes or baskets to preserve their quality (Tordecilla-Madera et al., 2018). Once these products have been delivered to customers, they are unpacked and RTIs are prepared to be returned to the supplier (Kim et al., 2014). The drink industry also uses RTIs in the distribution process to decrease costs or loss rates. This is the case, for instance, when transporting beer (Fan et al., 2019) or soft drinks (Soysal, 2016; Koç and Laporte, 2018). Finally, Mason et al. (2012) provide an example from the gas industry. These authors focus on tracking the cylinders since these RTIs are highly likely to be lost or stolen. They present an inventory management system based on the use of radio frequency identification (RFID) technology as a more sophisticated identification technique (Ilic et al., 2009). The problem discussed in this Section is motivated by a real-world case from the agri-food

industry: a Colombian company that produces packed bread and cereal, distributing these products in RTIs. For a complete review including several real-life cases on the use of RTIs, readers are referred to Mahmoudi and Parviziomran (2020).

Regarding RTIs management, three control strategies have been proposed by Kroon and Vrijens (1995): (i) the switch pool system, which is an integrated system between suppliers and customers where each one owns RTIs –the supplier is responsible for the return process in this strategy; (ii) systems with return logistics, in which RTIs are property of a central agency –this agency takes care of their collection; and (iii) systems without return logistics, in which RTIs are also a property of a central agency –here, suppliers rent only the required RTIs and are responsible for the return process. Finally, if some RTIs are not used, they can be returned to the agency. These strategies are analyzed by Hellström and Johansson (2010) through a case study, where the aim is to reduce the cost of RTI management and transport. A good strategy for RTIs management is the interchange between suppliers and customers (Elia and Gnoni, 2015), i.e., suppliers deliver products in RTIs and customers return them empty. However, such interchange cannot be done simultaneously because pickups are only possible when deliveries have already been made and the vehicle is empty (Koç and Laporte, 2018). Therefore, the return of RTIs has to be performed in two ways: (i) by using dedicated collection vehicles; and (ii) by using vehicles that make deliveries firstly and then collections. The latter case is known as the vehicle routing problem with backhauls (VRPB) (Berbeglia et al., 2007; Belloso et al., 2017). Being a rich extension of the well-known vehicle routing problem (Caceres-Cruz et al., 2015), the VRPB is also an *NP-hard* problem.

Studies on the VRPB typically assume that all customers must be visited. However, the scenario in which visiting backhaul customers is optional is explored in this Section. In Figure 3.11, the process with the optional collection of RTIs is depicted.

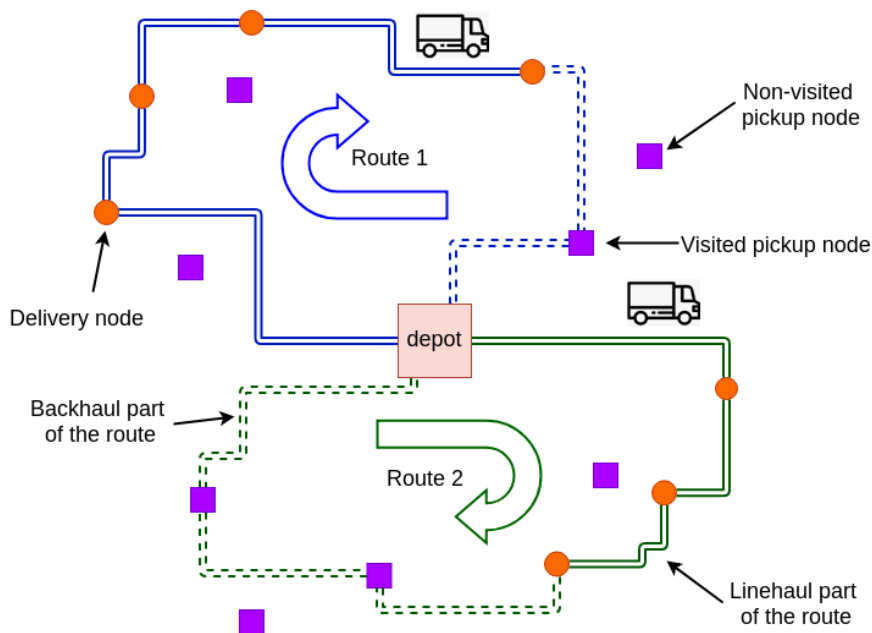


Figure 3.11: Illustrating the vehicle routing problem with optional backhauls.

As illustrated in Figure 3.11, the carrier might decide not to visit some backhaul nodes, thus incurring a penalty cost –which is associated with the fact that some customers will need to temporarily store the empty RTIs until a new visit is scheduled. Also, for instance, if the quantity of RTIs available in a supplier’s facility is enough to ensure future deliveries, the collection of RTIs in the current period can be reduced to diminish routing cost or speed up the routing process. It makes sense whenever the cost of holding the additional inventory at the customers’ facilities –i.e., the cost of holding the RTIs in stock– is lower than the marginal routing cost associated with their collection. In practice, the cost of holding RTIs in stock at the customers’ facilities might vary from one period to another, depending on factors such as how many RTIs are needed for the next period distribution or how long have been the RTIs staying at the customers’ inventories. Accordingly, the main contribution of this Section can be summarized as follows: (i) the vehicle routing problem with optional backhauls (VRPOB), which has been scarcely considered in the literature, is analyzed; (ii) a mathematical model of the problem, which is then solved using exact methods for small-scale instances, is proposed; (iii) a biased-randomized iterated local search is proposed to solve larger instances of the problem; and (iv) different levels of penalty cost are considered, and how routing solutions evolve for these levels is analyzed. Biased-randomization techniques allow for extending traditional metaheuristic frameworks in order to enhance their performance (Gonzalez-Neira et al., 2017; Ferone et al., 2018) and facilitate ‘agile’ optimization (Martins et al., 2020c).

3.2.1 Literature Review

Decisions regarding the integration of forwarding logistics with reverse logistics have been studied systematically over the last three decades. In the following, some works related to the vehicle routing problem with backhauls are reviewed. A highly cited survey on the topic is carried out by Parragh et al. (2008). They classify this problem in four groups: (i) the VRP with clustered backhauls (VRPCB), where linehaul (delivery) customers are served firstly and then backhaul (pickup) customers are visited for collecting the items that need to be returned to the depot; (ii) the VRP with mixed linehauls and backhauls, where the constraint in group (i) is not considered; (iii) the VRP with divisible delivery and pickup, where some customers demand both a delivery and a pickup and they can be visited twice; and (iv) the VRP with simultaneous delivery and pickup, where all customers must be visited once. The problem analyzed in this Section is strongly related to the first group, although the term VRPB, used by Goetschalckx and Jacobs-Blecha (1989) and Toth and Vigo (1997), is used to refer to this problem.

Some constraints that can lead to this type of sequential process are: (i) deliveries have priority over collections; (ii) due to time or space limitations, it is not possible to load returned products to a vehicle that still has products to deliver; or (iii) cross-contamination between products to deliver and returned products must be avoided. Traditionally, the main goal of the VRPB has been to minimize the total distribution and collection cost by taking advantage of the non-used capacity of the vehicles in the return trip. Some initial approaches for the VRPB are presented by Deif and Bodin (1984) and Jacobs-Blecha and Goetschalckx

(1992). Both papers present heuristic algorithms to solve the problem efficiently. Besides, the VRPB is also modeled as a mixed-integer linear problem by Toth and Vigo (1997). They solve it through a branch-and-cut algorithm for instances between 25 and 68 customers. As pointed out by Koç and Laporte (2018) and Beloso et al. (2019), the VRPB is still offering challenges that need to be solved, such as including stochastic parameters or using hybrid solution methods.

In Wassan (2007), a heuristic that uses reactive tabu search and adaptive memory programming is proposed for solving the VRPB. Zachariadis and Kiranoudis (2012) propose the static move descriptor strategy, which is intended to reduce the computational complexity required to examine neighborhoods with very large solutions. Brandão (2016) solves a VRPB through a deterministic version of the iterated local search metaheuristic. Dominguez et al. (2016a) consider two-dimensional loads in a VRPB. Here, a hybrid approach that combines biased-randomization with a large neighborhood search metaheuristic is proposed. In general, better solutions –in terms of cost and computational times– were found in comparison with current state-of-the-art heuristics. Beloso et al. (2019) use an iterative method based on local search and a biased-randomization process to solve the heterogeneous-fleet VRPB, obtaining 20 new best-known solutions in a set of 36 instances.

The VRPB with time windows (VRPBTW) for visiting customers has also been studied. For instance, Küçükoğlu and Öztürk (2015) propose a hybrid metaheuristic algorithm, which integrates simulated annealing with tabu search and a local search. A total of 34 best-known solutions were found for a benchmark set of 45 instances. A heterogeneous fleet is considered by Wu et al. (2016), who solve the VRPB with time windows using an algorithm based on ant colony optimization. Variables such as *vehicle type*, *fleet size*, and *routes* are determined in order to minimize the service total cost. Lin et al. (2017) integrate new operative constraints into the VRPB with time windows, among them: last-in-first-out rules, vehicle capacity depending on the specific order, driving-time limits, etc. To solve this version of the problem, a hybrid approach combining the greedy randomized adaptive search procedure with tabu search is proposed. A two-phase approach that combines tabu search with a multi-start evolutionary strategy is proposed by Reil et al. (2018) to solve a VRPBTW. These authors also consider constraints regarding three-dimensional loading and different backhaul strategies.

Regarding the consideration of inventories, a large number of works have addressed either the VRPB or the inventory routing problem (IRP) (Guler et al., 2018; Guler et al., 2020). Nevertheless, both problems have been rarely studied jointly. For instance, Arab et al. (2018) consider both problems by addressing the IRP with backhauls. The goal is twofold: to minimize the total cost (inventory plus transportation) as well as transportation risk. A heterogeneous fleet, multiple periods, and multiple products are considered. Exact algorithms using the ϵ -constraint method and evolutionary algorithms are applied to solve the problem.

All the aforementioned papers assume that visiting every pickup customer is mandatory. Nevertheless, the selection of customers to service is also an alternative, i.e., only some

pickup customers are visited. In the context of the general VRP (not the VRPB), this problem is called the VRP with Deliveries and Selective Pickups (VRPDSP) (Gribkovskaia et al., 2008). It can be classified into two groups:

1. Deliveries and pickups are carried out alternately. Most found papers belong to this category. Different characteristics are tackled, e.g., multiple objectives (Assis et al., 2013), a single-vehicle (Gribkovskaia et al., 2008), multiple vehicles (Ting et al., 2017), customers can be visited more than one time, both for deliveries and for pickups (Del Ser et al., 2017), different nodes for origin and destination (Ting and Liao, 2013), only one pickup customer must be serviced (Falcon et al., 2010), and time windows (Al Chami et al., 2018).
2. Pickups must be carried out only after all deliveries are done, i.e., this problem shows characteristics of a VRPB. To the best of our knowledge, only García-Nájera et al. (2015) address backhauls as a central problem. Bruck and Iori (2017), and Gutiérrez-Jarpa et al. (2010) also tackle backhauls, but only for comparative purposes with other problems. García-Nájera et al. (2015) consider a multi-objective problem, Bruck and Iori (2017) consider single-vehicle routes, and Gutiérrez-Jarpa et al. (2010) use an exact algorithm. Besides, none of these papers consider the concepts of penalty costs and RTIs. By including these realistic characteristics, this work goes one step ahead in the state-of-the-art.

Most of the reviewed examples belong to either the VRPB class or the VRPDSP class. The former addresses those cases in which all customers are serviced and all deliveries have been completed before any collection is considered. The latter gives the chance of not visiting some pickup customers. This work takes characteristics of both classes and analyzes the vehicle routing problem with optional backhauls, in which some pickup customers may not be visited, and all deliveries are always completed before visiting pickup customers.

3.2.2 Problem Definition

The vehicle routing problem with optional backhauls consists of a set of linehaul (LH) and backhaul (BH) customers whose demands must be satisfied –at least the LH ones– using a fleet of homogeneous vehicles that are initially located at a depot. This depot has enough capacity and vehicles to cover the aggregated customers' demand. Therefore, all LH customers will be serviced. The supplier uses returnable transport items for transporting product units. Hence, after delivering all units, empty RTIs from previous deliveries should be collected at some BH customers and returned to the central depot for future deliveries. Collecting RTIs from BH customers is optional, and not doing it might generate savings in transportation costs, but it will also raise some 'penalty' costs associated with the lack of service –which implies that some customers will need to keep the RTIs in stock until the next visit. Thus, in the VRPOB, the following decisions need to be made in order to minimize the total cost (routing cost plus penalty cost): (i) to determine which backhaul customers will be visited;

(ii) to assign customers to a predefined number of routes (and vehicles), and (iii) to establish the sequence in which customers should be visited.

Consider a non-directed graph $G = (V, A)$, with $V = \{0\} \cup L \cup B$ representing the set of nodes, being node 0 the depot, $L = \{1, 2, \dots, n\}$ the set of n linehaul customers, and $B = \{n + 1, n + 2, \dots, n + m\}$ the set of m backhaul customers. Likewise, $A = \{(i, j) : i, j \in V, i < j\}$ is the set of edges linking each pair of nodes. For each $i \in L$, there is a positive demand $d_i > 0$ representing units of product to be delivered. Similarly, for each $j \in B$, there is a negative demand $d_j < 0$ representing items to be collected. Also, a set K of homogeneous vehicles available at the depot is considered, being each of them with a capacity $q \gg \max\{|d_i| : i \in V\}$. Travelling an edge from node i to node j has a cost $c_{ij} = c_{ji} > 0$. The following constraints need to be considered:

- Each route begins and ends at the depot.
- Each route must have at least one LH customer, i.e.: routes formed just by BH customers are not allowed.
- In any route, LH customers are serviced before BH customers.
- Each LH customer must be serviced, but visiting BH customers is optional.
- For each route, the quantity of product to deliver and to collect must not exceed the vehicle's capacity.
- Whenever a LH customer is visited in a route, all its demand is serviced; similarly, whenever a BH customer is visited in a route, all its items are collected.

Accordingly, the complete mathematical formulation for this problem (Londoño et al., 2020) can be found in Appendix B.2 (Section B.2.1).

3.2.3 Solution Method: a Biased-Randomized Iterated Local Search

Being an extension of the vehicle routing problem, the VRPOB is also an *NP-Hard* problem. Therefore, a biased-randomized version of a metaheuristic algorithm is proposed to solve large-size instances of the VRPOB. This algorithm combines concepts from the ILS framework (Lourenço et al., 2010) and the iterated greedy framework (Ruiz and Stützle, 2007). Hence, the algorithm proposed in this Section relies on a perturbation stage (a destruction-reconstruction one), which significantly modifies a base solution, followed by a local search that tries to improve the modified solution. The algorithm also uses an acceptance criterion to update, whenever the criterion is satisfied, the base solution even if the new solution does not improve it. This whole process is done iteratively. Algorithm 10 presents the general idea of this approach, henceforth named as *BR-ILS*. Firstly, an initial solution is obtained through a savings-based heuristic (line 1), which incorporates a penalization strategy to delay merging between LH and BH nodes. This initial solution is improved through a fast local search procedure (line 2) composed of: (i) node-insertion and node-swap operators; and (ii) a cache (hash map) data structure. Then, the iterated search starts from this

initial solution, henceforth called as *base* solution. Here, $r\%$ of the routes from the base solution are destroyed and reconstructed (line 5). It implies solving a smaller problem using a biased-randomized version of the heuristic (line 6), which is later incorporated into the base solution (line 7). This new solution is later improved by applying a local search procedure (line 8). Finally, an acceptance criterion –based on the cost of the solutions– is employed for selecting and evaluating the best-found and base solutions (line 9). This process is repeated until an ending criterion is met (line 10). Finally, the best-found solution is returned (line 11). These steps are detailed below.

Pseudocode 10: Biased-Randomized Iterated Local Search

Data: problem input parameters $inputParameters$, balance factor between penalty costs and traditional savings α , delaying factor of interface edges λ , geometric distribution parameter β , destruction ratio r

1 **Function** BR-ILS($inputParameters, \alpha, \beta, \lambda, r$):

2 $init_{sol} \leftarrow \text{penalizedBRSavingsHeuristic}(inputParameters, \alpha, \beta, \lambda)$

3 $init_{sol} \leftarrow \text{fastLocalSearch}(inputParameters, init_{sol})$

4 $base_{sol} \leftarrow init_{sol}$

5 **while** *time does not reaches the limit* **do**

6 $pendingNodes \leftarrow \text{destroyRoutes}(r, base_{sol})$

7 $sub_{sol} \leftarrow \text{penalizedBRSavingsHeuristic}(inputParameters, \alpha, \beta, pendingNodes)$

8 $new_{sol} \leftarrow base_{sol} \cup sub_{sol}$

9 $new_{sol} \leftarrow \text{fastLocalSearch}(inputParameters, new_{sol})$

10 $best_{sol} \leftarrow \text{acceptanceCriterion}(base_{sol}, new_{sol})$

11 **end**

12 **return** $best_{sol}$

13 **End**

3.2.3.1 Generating an Initial Solution

In order to generate an initial solution, a savings-based heuristic is enhanced and extended by including the following three strategies: (i) introducing a penalization cost in the savings calculation, which refers to the possibility of considering optional backhauls; (ii) penalizing any merging process between LH and BH nodes to delay their selection –this strategy guarantees that all deliveries are done before pickups; and (iii) including a biased-randomized procedure during the search stage. The first novelty of this approach refers to an alternative savings value for merging routes. Since each BH customer has a penalty cost for not being served, the traditional way to calculate such savings ($s_{ij} = c_{i0} + c_{0j} - c_{ij}$) is extended according to the expression in Equation (3.8), proposed by Panadero et al. (2020a) for the team orienteering problem –notice that this extended savings applies just to edges whose both nodes are BH ones.

$$s'_{ij} = \alpha * s_{ij} + (1 - \alpha) * (h_i + h_j), \forall i \in B, \forall j \in B \quad (3.8)$$

In Equation (3.8), $0 \leq \alpha \leq 1$. The idea is that this α tries to balance both penalty costs and traditional savings in distances or times. The case in which $\alpha = 1$ corresponds to the traditional case in which penalty costs are not considered. In the case in which $\alpha = 0$, savings in

transport cost are not considered, and only penalty costs are relevant. Intermediate values of α assign more or less weight to transport and penalty costs. As a second extension, saving values are updated in order to address properly the backhaul nodes and, consequently, provide a feasible merging between LH and BH customers. This procedure is described in Algorithm 11. The idea is to delay the selection of interface edges, i.e., those ones linking a LH node with a BH node, so that both LH and BH routes are “complete” before being merged. This delay is done by subtracting p from the previously computed savings (line 7). The value of p is computed as $p = max_s * \lambda$ (line 6), in which max_s represents the maximum savings that can be attained (line 2), while λ is a penalty coefficient ranging between 0 and 1. In this case, λ is uniformly chosen at random between 0.05 and 0.20 (line 5).

Pseudocode 11: Generating the Penalized Savings List

Data: problem input parameters `inputParameters`, balance factor between penalty costs and traditional savings α , delaying factor of interface edges λ

```

1 Function penalizeSavingsList(inputParameters,  $\alpha$ ,  $\lambda$ ):
2   savingsList  $\leftarrow$  createSavingsList(inputParameters,  $\alpha$ )
3    $max_s \leftarrow$  obtainMaximumSavings(savingsList)
4   foreach edge in savingsList do
5     if edge is interface then
6       Randomly select  $\lambda \in \{0.05, 0.2\}$ 
7        $p \leftarrow max_s * \lambda$ 
8       updateSavings(edge,  $p$ )
9     end
10  end
11  return savingsList
12 End

```

The first stage of constructing an initial solution refers to the creation of a dummy solution, which is composed of a set of single-node routes. Therefore, the last extension in this stage is the use of BR techniques (Grasas et al., 2017) to guide the selection of an element in the penalized savings list. BR assigns a probability of being chosen to each edge in this list. Therefore, the selection is smoothed by replacing the original greedy behavior with a probabilistic one. To achieve this purpose, the geometric probability distribution is employed in a MCS, in which only one parameter (β) must be fine-tuned. After sorting the savings list in descending order, β can be interpreted as the probability of choosing the edge with the highest savings. As a result of performing some preliminary experiments, we observed that good results are obtained when β is selected uniformly randomly between 0.01 and 0.50. The selection within this interval provides the algorithm with the right balance between exploration and exploitation of the solution space, since $\beta = 0$ refers to a uniform random selection, and $\beta = 1$ represents a greedy strategy.

Once a promising merging edge is selected, a set of pre-defined merging conditions are checked. The two corresponding routes of an edge e can be merged if: (i) its nodes are adjacent to the depot; (ii) its nodes belong to different routes; and (iii) the vehicle capacity constraint is met. If both routes belong to the same cluster, i.e., to either the LH group or the BH group, the merge is accepted. Otherwise, the optional pickups for that merging is analyzed: this union is allowed only if total transportation cost is less than total penalty cost

for not picking up. Thus, if the total transportation cost for BH nodes is high, the algorithm is more likely to choose not to collect RTIs in that route. In this case, the union is discarded, and the BH route is penalized. At the end of each iterative step, the selected edge is removed from the savings list, and the process is repeated until no more edges are available.

By not taking into account the number of vehicles at this stage, this procedure for creating a solution might lead to infeasible solutions regarding this hard constraint. Therefore, a recursive corrective operator is executed at the end of the procedure to attain it (Belloso et al., 2017). Broadly speaking, this procedure relies on selecting potential routes to be kept in the solution –according to its current status– and reconstructing new routes in order to meet the number of routes constraint. When the current solution contains fewer routes than required (the available number of vehicles), large-sized demand routes are more likely to be chosen to be incorporated into the new solution, then allowing the construction of more routes. On the other hand, when the solution is composed of more routes than the number of available vehicles, small-sized demand routes have more probability of being selected. The remaining nodes –from non-selected routes– are now part of a sub-problem, which is solved by the same mechanism, and then incorporated into the final solution. This procedure is repeated until a feasible solution –i.e., a solution which is composed of $|K|$ routes– is found.

3.2.3.2 Local Search Procedures

The procedure in Section 3.2.3.1 yields an initial solution that can be improved. Taking into account the problem structure, a local search (LS) procedure based on the use of an inter-routes node-insertion operator, an inter-routes nodes-swap operator, and a cache (memory-based or hash map) data structure is implemented. The cache procedure relies on storing in memory the best-found route for a given group of nodes (Juan et al., 2011). Whenever a lower-cost route formed by the same set of nodes is found, it is returned by the method. Otherwise, in case of the current route has a better cost, it replaces the existing one in memory. Finally, if the route does not exist, it is inserted into the cache. This procedure is applied whenever a new solution is constructed (Algorithm 10, lines 2 and 8).

3.2.3.3 Perturbation Stage

The improved initial solution is used as a base solution by the algorithm. At each iteration, this base solution is perturbed. The perturbation process consists in destroying at least two routes (Algorithm 10, line 5) and reconstructing them. Such destruction implies that, temporarily, m nodes do not belong to any route. Observe that $m < n$, where n is the total number of nodes in the instance. This implies that a sub-problem must be solved and, therefore, a smaller solution will be attained. In this way, a destruction ratio r is defined in order to establish a maximum number of routes to be destroyed (DR). Following Belloso et al. (2017), r is generated as a random number between 0.10 and 0.50. According to these authors, a general value was adopted for this parameter in order to avoid the need for performing complex and time-expensive fine-tuning processes. On the one hand, a destruction

ratio larger than 0.50 would destroy most of the constructed routes and the process would lose time-efficiency. On the other hand, a destruction ratio lower than 0.10 would destroy very few routes, and the effect of the perturbation would be almost negligible. The number of routes to be destroyed is computed as the maximum between 2 routes and an $r\%$ of the total routes in the solution:

$$DR = \max\{2, r * \#Routes\} \quad (3.9)$$

Once the destruction process is finished, the reconstruction is made from scratch for the m nodes, by using again the procedure described in Section 3.2.3.1. After a feasible solution is generated, the local search mechanism described in Section 3.2.3.2 is applied (Algorithm 10, line 8), and the acceptance criterion is tested (Algorithm 10, line 9). This process (Algorithm 10, lines 4-10) is repeated until a maximum time limit is reached.

3.2.3.4 Acceptance Criterion

An acceptance criterion is incorporated into this approach in order to accept promising solutions and avoid local minima. This mechanism relies on the possibility of updating the base solution by another one of lower quality in order to provide the exploration of new regions in the solution space.

3.2.4 Computational Experiments and Results

For evaluating the performance of the *BR-ILS*, a set of 33 instances, originally introduced by Toth and Vigo (1997), was solved. These instances differ in the number of LH and BH nodes, vehicles capacities, and demands. In total, 3 proportions of LH customers are used: 50%, 66%, and 80%. Thus, for instance, in the latter case (80% or 4 out of 5) customers 1, 2, 3, and 4 are LH, while the 5th one is BH, and so on. Here, it is assumed that RTIs to be collected were used in previous periods to deliver products to the associated customer. While small instances can be solved through an exact method, larger instances need a heuristic or meta-heuristic approach such as the one introduced in Section 3.2.3. The proposed approach has been implemented in Java, and a standard PC with an Intel Core *i7* CPU at 2.7 GHz and 16 GB RAM has been employed to run all tests.

Initially, it is assumed that visiting all customers is mandatory (i.e., for any customer, the penalty cost associated with non-visiting it is extremely large, i.e., $h_i = \infty$). This allow us to compare the performance of the proposed algorithm with the results provided in Toth and Vigo (1997) and Bellosso et al. (2017). Table 3.6 shows the comparison between our best solutions (OBS) and the best-known solutions (BKS) in the literature. For each problem instance, the number of LH and BH nodes, vehicle capacity Q , number of vehicles K , and the solutions' cost are presented. Notice that, the proposed method is able to match the BKS for 91% of the tested instances, also achieving similar results to the ones provided by Bellosso et al. (2017) and outperforming the results in Toth and Vigo (1997). For the cases in which Bellosso et al. (2017) achieve better solutions, the gap never exceeds 0.5%. Therefore, it is possible to conclude that the *BR-ILS* algorithm obtains competitive solutions when compared with

state-of-the-art approaches in the VRPB with mandatory backhauls. The next step is to use the proposed algorithm to solve the VRPOB, that is, the version of the problem in which visiting backhaul customers is optional and, therefore, can be skipped if savings in routing cost are higher than the associated penalty cost.

Table 3.6: Comparison between our algorithm and previous works.

Instance	LH	BH	Q	K	Toth and Vigo (1997) (BKS in 1997)	Belloso et al. (2017) (BKS in 2017)	OBS ($h_i = \infty$)	GAP OBS-Belloso
eil22_50	11	10	6000	3	371.0	371.0	371.0	0.0%
eil22_66	14	7	6000	3	366.0	366.0	366.0	0.0%
eil22_80	17	4	6000	3	375.0	375.0	375.0	0.0%
eil23_50	11	11	4500	2	682.0	682.0	682.0	0.0%
eil23_66	15	7	4500	2	649.0	649.0	649.0	0.0%
eil23_80	18	4	4500	2	623.0	623.0	623.0	0.0%
eil30_50	15	14	4500	2	501.0	501.0	501.0	0.0%
eil30_66	20	9	4500	3	537.0	537.0	537.0	0.0%
eil30_80	24	5	4500	3	514.0	514.0	514.0	0.0%
eil33_50	16	16	8000	3	738.0	738.0	738.0	0.0%
eil33_66	22	10	8000	3	750.0	750.0	750.0	0.0%
eil33_80	26	6	8000	3	736.0	736.0	736.0	0.0%
eil51_50	25	25	160	3	559.0	559.0	559.0	0.0%
eil51_66	34	16	160	4	548.0	548.0	548.0	0.0%
eil51_80	40	10	160	4	565.0	565.0	565.0	0.0%
eilA76_50	37	38	140	6	739.0	739.0	739.0	0.0%
eilA76_66	50	25	140	7	768.0	768.0	768.0	0.0%
eilA76_80	60	15	140	8	781.0	781.0	781.0	0.0%
eilB76_50	37	38	100	8	801.0	801.0	801.0	0.0%
eilB76_66	50	25	100	10	873.0	873.0	873.0	0.0%
eilB76_80	60	15	100	12	919.0	919.0	919.0	0.0%
eilC76_50	37	38	180	5	713.0	713.0	713.0	0.0%
eilC76_66	50	25	180	6	734.0	734.0	734.0	0.0%
eilC76_80	60	15	180	7	733.0	733.0	733.0	0.0%
eilD76_50	37	38	220	4	690.0	690.0	690.0	0.0%
eilD76_66	50	25	220	5	715.0	715.0	715.0	0.0%
eilD76_80	60	15	220	6	703.0	694.0	695.0	0.1%
eilA101_50	50	50	200	4	843.0	831.0	831.0	0.0%
eilA101_66	67	33	200	6	846.0	846.0	846.0	0.0%
eilA101_80	80	20	200	6	916.0	856.0	856.0	0.0%
eilB101_50	50	50	112	7		923.0	925.0	0.2%
eilB101_66	67	33	112	9		982.0	987.0	0.5%
eilB101_80	80	20	112	11		1008.0	1008.0	0.0%
<i>Average</i>								0.0%

Four scenarios are considered for the unitary penalty cost, h_i , namely: *high*, *medium-high*, *medium-low*, and *low*. The values for these h_i scenarios were established according to the demand size of the instances. Accordingly, new tests were carried out in order to measure the performance when parameters h_i and α , from Equation (3.8), are incorporated in the solution cost. The parameter α is set as $\alpha \in \{0.2, 0.5, 0.8, 1.0\}$. Regarding the unitary penalty costs, $h_i \in \{0.66, 0.33, 0.16, 0.06\}$. Specifically, for h_i , this parameter was set from higher to lower values in order to measure the impact on the cost when collecting and not collecting the RTIs, respectively. A higher value of h_i ensures not to fail to collect any RTI, while a lower h_i might generate a better solution cost by not collecting all of them. For calibrating

the parameter α , the methodology proposed by Calvet et al. (2016b) was followed, who provided a general procedure, based on statistical learning. This procedure does the following: (i) it chooses a subset of benchmark instances at random; (ii) it selects the range over which each parameter will be varied; (iii) it applies an experimental design to explore promising regions; and (iv) it obtains a set of parameter values by intensifying the search. Following this procedure, only small instances with known optimal solutions were considered. Thus, for each combination of instances, unitary penalty cost h_i , and α , a total of 30 runs were performed (each run using a different seed for the pseudo-random number generator). The performance of the metaheuristic was measured as the percentage gap between OBS and the BKS –which is also the optimal solution in this case. After these experiments, a value of $\alpha = 0.8$ is set for testing our approach on large-sized instances.

With the value of parameter alpha defined, a total of 30 runs were performed for each of the 33 instances and h_i value. In the case of large instances (those ones composed of 55 nodes or more), a new fine-tuning process was carried out to establish their h_i values, namely: high ($h_i = 12.40$), medium-high ($h_i = 6.20$), medium-low ($h_i = 3.10$), and low ($h_i = 1.24$). Re-adjusting the values of h_i for these instances is necessary since they have different levels of demand as compared to the smaller ones. The stopping criterion was fixed to 25, 75, and 300 seconds for instances up to 50 nodes, up to 100 nodes, and with over 100 nodes, respectively. Tables 3.7 and 3.8 show the results obtained by the *BR-ILS* algorithm. For each instance and h_i level, the following data is provided: the BKS (from the literature, considering the case in which the collection of RTIs is mandatory), OBS, optimal solution cost, average cost (AVG) and standard deviation (SD), the number of non-serviced BH customers (NS), CPU time (in seconds), and the percentage gap. The optimal solutions were obtained by Londoño et al. (2020) through the CPLEX solver. Therefore, we provide two different comparisons: between OBS and BKS (when the RTIs collection is mandatory) and between OBS and CPLEX (when the collection is optional).

As we can notice, the results obtained for small instances (22, 23, 30 and 33 nodes) are the same as the optimal ones for 45 out of 48 instances. Besides, the times employed by the proposed approach are much smaller than the ones employed by the exact method. For the CPLEX, computational times increase dramatically with the size of each instance. For example, instance *eil51_80*, from Table 3.8, employs more than 6.5 hours in finding the optimal solution. More relevant results are obtained when the number of non-serviced BH customers is greater than zero. Here, 45 solutions –from both Tables 3.7 and 3.8– show this effect. In 43 out of these (93%), the gap is negative, meaning that, by allowing collections to be optional, it is possible to generate solutions with a lower total cost than when all BH customers need to be visited. Regarding the variance of the results, the columns AVG (SD) show a relatively small dispersion around the average cost, which allows us to illustrate the robustness of *BR-ILS* methodology.

Based on Tables 3.7 and 3.8, Figures 3.12 and 3.13 present a boxplot of the percentage gaps between OBS and the BKS, as well as of the number of non-collected RTIs for each penalization level. Here, we are comparing our results with the BKS in order to verify the benefits of the optional collection of RTIs. Notice that, for smaller penalization values, the

solutions present a significant reduction in transportation costs, achieving a reduction of about 25%. By analyzing Figure 3.13, one can observe the following: as the penalization value decreases, the number of uncollected RTIs of the solution increases. Therefore, relaxing the constraint of having to visit all customers can bring significant savings in the routing cost and, consequently, provide better overall distribution plans for decision-makers.

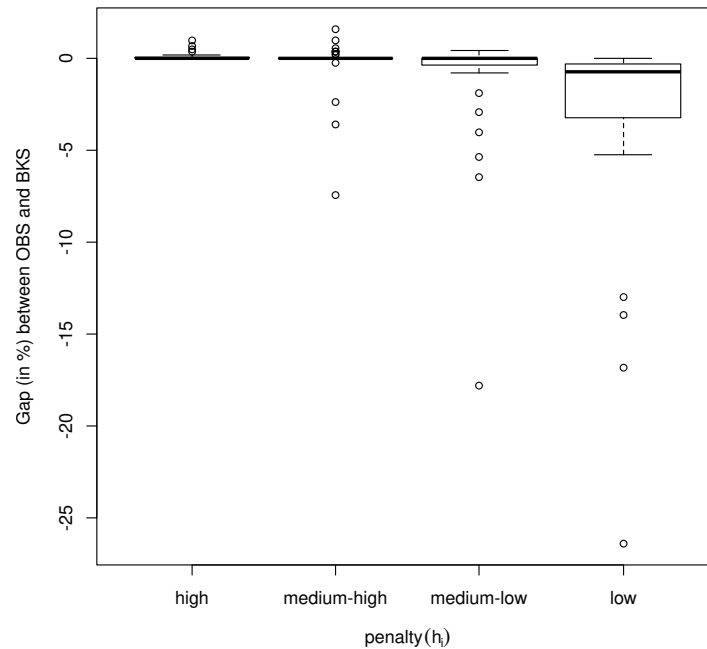


Figure 3.12: Gap between our OBSs and the BKSs for each penalization scenario (h_i).

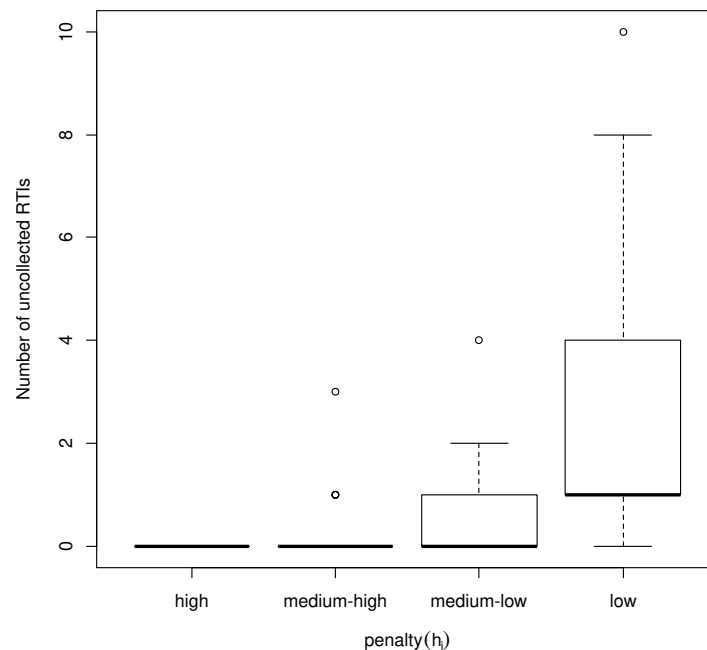


Figure 3.13: Number of non-collected RTIs of our OBSs for each penalization scenario (h_i).

There is not a general relation between the non-serviced BH customers and the gap.

However, in those cases in which a gap is positive, there are only 1 or 2 non-serviced BH customers. That is, a higher quantity of non-serviced BH customers usually yields savings in total cost. This does not mean that the RTIs will never be collected, but just that they can be picked up during the next time period. Likewise, BH customers that are visited during the current period might be skipped in the next one. The unitary penalty cost is also a parameter that influences cost savings. On the one hand, high values of h_i lead to the collection of most RTIs. In this case, we obtain an average gap of 0.0%, which highlights the efficiency of the proposed solving approach. On the other hand, only in a few cases a very low h_i does not yield savings, which shows the advantage of using our approach for such values of h_i .

The following analysis aim at comparing the case in which deliveries and pickups are made independently, i.e., LH customers and BH customers are not serviced in the same route but in different routes, with the integrated strategy. Table 3.9 shows the associated costs of both independent and integrated deliveries and pickups. As we can see, the convenience of merging the linehaul and backhaul routes generates noticeable reductions in cost. By integrating them in the same route, it is possible to reduce transportation costs between 20.8% and 55.3%.

From a managerial perspective, it has been shown that, if the traditional hard constraint of visiting all customers is relaxed, better overall distribution plans can be found in scenarios with a moderated penalty cost. Hence, it is interesting to analyze how the total transportation cost varies as the unitary penalty cost (h_i) increases or decreases. This transportation cost depends on the value of h_i , since this value affects both the number of unvisited backhaul customers as well as the number of uncollected RTIs. When h_i is relatively high, all backhaul customers are visited. However, as h_i decreases some RTIs are not collected (since the cost of visiting those customers is higher than the penalty cost associated with not visiting them). Therefore, the decision of not visiting some customers is strongly influenced by the specific value of h_i . Of course, when addressing real-world cases, decision-makers must estimate their own penalty costs, but our methodology can still be used with the specific data of each case.

3.2.5 Conclusions

Setting delivery and collection routing plans represents a challenging problem that must be addressed daily in many supply chains. In this Section, a variant of the vehicle routing problem with linehaul and backhaul customers was analyzed, in which visiting the latter is an optional action that might be skipped. If so, savings in transportation costs are achieved, although penalty costs are incurred. These penalties can be understood as additional inventory costs for holding returnable transport items (RTIs), which are located at the customer facilities. A real-world case from the Colombian agri-food industry motivates our study. The experience in this company indicates that RTIs recovery is an activity that allows for preserving a valuable asset. If the recovery is successful, RTIs may be re-used many times to carry out new deliveries, i.e., it is assumed that RTIs to collect were used to distribute products in a previous period.

Two approaches are proposed to deal with this *NP-Hard* combinatorial optimization problem: (i) a mixed-integer linear programming model, which is solved for a set of traditional instances. Here, instances up to 51 nodes were solved using exact methods; and (ii) a biased-randomized iterated local search for solving larger instances. In a preliminary stage, a set of computational experiments have been carried out in order to solve the traditional vehicle routing problem with backhauls. An average gap of 0% was achieved when comparing our results against an existing state-of-the-art approach in the literature. Moreover, the proposed approach is able to find all known optimal solutions in a few seconds against the exact approach, which requires up to approximately 7 hours to find the best solution for a medium-size instance. After showing that this approach is competitive with state-of-the-art methods to solve this problem, the proposed algorithm was adapted to consider that RTIs collection is optional, subject to a penalty (inventory holding) cost. Consequently, a new parameter α is introduced, which indicates the weight given to the transport cost versus the penalty cost. After calibrating α , the corresponding analysis shows that it is possible to obtain lower aggregated (routing plus inventory) costs in scenarios where the collection is not mandatory. It was demonstrated that the collection decision is sensitive to the unit penalty cost h_i , i.e., tests show that the lower h_i , the greater the number of not collected RTIs. This action yields substantial savings in total costs up to 26%. Nevertheless, some benchmark instances required visiting all customers regardless of the value of h_i . In other words, cost savings are instance-dependent, hence, decision-makers in real-world supply chains should estimate h_i as accurately as possible to decrease the total aggregated cost successfully.

Several lines are identified for future research: (i) our approach could be extended to consider the multi-period and / or multi-depot cases; and (ii) customers' demands or (time-based) transportation costs might be considered as random variables.

Table 3.9: Comparison between VRPB and independent deliveries and pickups.

Instances	LH	BH	Independent deliveries and pickups			VRPB	% Cost reduction
			Deliveries cost	Pickups cost	Total cost	Total cost	
eil22_50	11	10	281	225	506	371	36.4%
eil23_50	11	11	397	434	831	682	21.8%
eil30_50	15	14	328	340	668	501	33.3%
eil33_50	16	16	633	513	1146	738	55.3%
eil51_50	25	25	340	335	675	559	20.8%

Chapter 4

Applications in Humanitarian Logistics

This chapter ¹ studies a Two-Echelon Vehicle Routing Problem with Pick-up and Delivery, which is addressed in the context of humanitarian logistics. In this case, a set of pharmaceutical laboratories holds a limited inventory of drugs needed in the disaster areas and must be served with raw material from a single distribution center or depot. On the other hand, these intermediate facilities must serve as fast as possible a set of final delivery points in the affected area. Due to the resulting poor transportation infrastructure after a disaster event, the affected areas might become no longer accessible by conventional cargo vehicles. Therefore, a single fleet of drones is employed in both delivery levels. To solve this 2E-VRP, an ‘agile optimization’ approach –which combines a biased-randomized algorithm with parallel computing– is proposed. This solving method is able to generate feasible solutions violating the real-time constraint.

4.1 The Two-Echelon Vehicle Routing Problem with Pick-up and Delivery

Real-time optimization, where decisions need to be made in just a few seconds or even milliseconds, has many application areas in logistics. For example, in the event of disasters, real-time optimization can be a life-saving differential. In this context, last-mile distribution logistics is related to the delivery of urgently needed goods to areas where roads are blocked by extreme weather events, disasters, or traffic congestion.

A motivational example is the distribution of drugs with drones in disaster situations. When disasters occur, an effective system of drug management should be established by health agencies in order to: (i) ensure the efficient, cost-effective, and rational use of the drugs; (ii) prevent and reduce excess mortality and morbidity; and (iii) promote a return to normalcy (McConnan, 2004). This distribution process is based on selection, procurement,

¹The contents of this chapter are based on the following work:

- Martins, L. C.; Hirsch, P.; Juan, A. A. (2020): [Agile optimization of a two-echelon vehicle routing problem with pickup and delivery](#). *International Transactions in Operational Research*, 28(1), 201-221.

distribution, and use of pharmaceuticals in primary health care (Quick et al., 1997). This Section focuses on the distribution stage of this process, in which urgently-needed items (e.g., drugs) must be delivered to an affected area after a disaster occurred. These items should be delivered from a near pharmacy or laboratory (intermediate node) as fast as possible. On the other hand, these intermediate facilities should be served from a central warehouse or depot which holds raw material used to fabricate the processed items requested by the final users.

Due to the resulting poor transportation infrastructure after a disaster event, the affected areas might become no longer accessible by conventional cargo vehicles. Therefore, the use of drones can be seen as an effective way of delivering life-saving treatments directly to disaster locations. Examples of drone delivery applications can be found in health care delivery, which includes the safe delivery of medicines, vaccines, defibrillators, blood samples, disease test samples, and test kits in remote areas out of reach (Scott and Scott, 2017; Balasingam, 2017).

As aforementioned, in city logistics, a variant of the classical and well-known VRP is 2E-VRP, which can be found in several transportation systems. This multi-level distribution system combines two delivery levels, in which the first one addresses the delivery from the depot to intermediate facilities, while the second level regards the delivery from these intermediate facilities to final customers. In the context of this Section, a set of intermediate facilities (e.g., pharmaceutical laboratories) holds a limited inventory of drugs needed in the disaster areas and must be served with raw material from a single distribution center or depot. On the other hand, these intermediate facilities must serve as fast as possible a set of final delivery points in the affected area. The same fleet of drones is employed in both delivery levels. Although the motivational example for this study is the drug distribution in disaster circumstances, similar problems can be found in other situations, too.

To solve this problem in 'real-time,' i.e., a few seconds or even milliseconds, the concept of "agile optimization", which refers to the massive parallelization of a BR version of a constructive heuristics, is introduced. By using parallel computing to solve real-life VRPs, the resulting methodology is able to provide in milliseconds 'good' solutions to medium- and large-sized instances (Juan et al., 2013b). The use of BR techniques facilitates the design of powerful algorithms that can effectively be used to provide real-time solutions in a range of situations that arise in dynamic and emergency contexts (Ghiani et al., 2003).

4.1.1 Literature Review

Drones were initially used in military applications. However, their use in transportation has become a challenging trend in supply chains, in which companies from different industries have invested in the delivery of goods, including food and medical products (Bamburly, 2015). When designing a system for delivering drugs by using drones, several aspects from the application and physical limitations should be taken into account. From a physical point of view, Gatteschi et al. (2015) provided a detailed overview of these aspects applied to drug deliveries. Limitations such as battery, velocity, and weight, were pointed out by Wan et al. (2018), who proposed a new mechanism designed to safely transport medical aids to

the target area. From the application context, current and potential applications in health care are discussed by Balasingam (2017), who describes regulatory limitations and future innovations in drone technology, such as diagnostic capabilities by using telemedicine to patients in hard-to-reach areas.

Although the use of drones for commercial purposes represents a significant advance in logistics, many technological and regulatory obstacles must be overcome. Several worst-case results regarding the use of drones in VRPs have been proposed by Wang et al. (2017b). These worst cases reveal the benefits (amount of time that could be saved) of using drones combined with trucks instead of using only a fleet of trucks. Later, the same authors extended their previous work by explicitly considering limited battery life and cost objectives (Poikonen et al., 2017). Daknama and Kraus (2017) also studied the use of trucks and drones to deliver packages. In this case, the authors limited the number of packages that can be transported by drones at a time and imposed the return to a truck in order to charge their battery after each delivery. Both studies concluded that combining drones with trucks allows the truck to parallelize tasks, which represents a substantial improvement that must be considered in delivery systems. Dorling et al. (2017) proposed two multi-trip VRPs for drone delivery, which minimize costs subject to a delivery time limit and minimize the overall delivery time subject to a budget constraint, respectively. A cost function that considers the energy consumption model and drone reuse was proposed and incorporated in a SA heuristic for finding near-optimal solutions to practical scenarios. According to the results, the reuse of drones and the optimization of battery size results in a substantial improvement in drone delivery VRPs.

The use of parallel computing is a breakthrough that allowed the resolution of larger and complex optimization problems (Migdalas et al., 2013). Its combination with optimization has made it possible to design powerful algorithms that can effectively be used to provide real-time solutions in dynamic contexts (Ghiani et al., 2003). Several parallel and distributed computing approaches have been already applied to different VRP variants. One of the studies regarding the use of parallel and distributed computing for solving VRPs in real-time has been written by Juan et al. (2013b). The authors pointed out potential applications of distributed computing to solve large-size VRPs with real-life constraints. They proposed a solution approach, which combines parallel computing, simulation, and a BR heuristic for solving the VRP with stochastic demands. Recently, Rey et al. (2018) proposed a hybrid methodology based on ACO and local search procedures. Different levels of parallelism were tested, from the construction of TSP routes to the construction of a complete VRP solution from each TSP route. The parallel computing power was employed to generate high-quality solutions for the VRP.

4.1.2 Problem Definition

The problem addressed in this Section is based on the description presented by Abdulkader et al. (2018), previously introduced in Section 3.1.2. In the current application context, the depot holds raw materials, such as chemical and natural products, required by pharmaceutical laboratories (PL) in order to manufacture drugs to be delivered to final points. These

final points can be seen as first-aid locations (FAL) in an area affected by a disaster. Each FAL is assigned to emergency staff (doctors, nurses, etc.), who requires these processed items to attend the population in an emergency situation. The delivery in both, PLs and FALs, is done by drones in order to allow the transportation of goods to non-accessible areas, where conventional cargo vehicles cannot arrive by land. Sensors at PLs and FALs report in case of a disaster if a location is affected or not. They also give information about the impact of the disaster. This provides the depot with a situation picture of the affected areas. Needs for drugs are known in advance on the basis of an existing emergency plan in case of disasters. Authorities must quickly react to this new information in order to guarantee a fast delivery service. The affected area is a subset of the complete space, which might include some PLs. In this case, these facilities are treated as FALs which require their specific medical supplies from an available non-affected intermediate facility. Figure 4.1 provides an example of a scenario in which sensors in the affected area communicate with the depot when a disaster occurs. This area is composed of FALs and PLs. Each location is associated with a sensor. In this example, all affected nodes (including FALs and PLs) must be served from the three available PLs that, on the other hand, must be served from the depot.

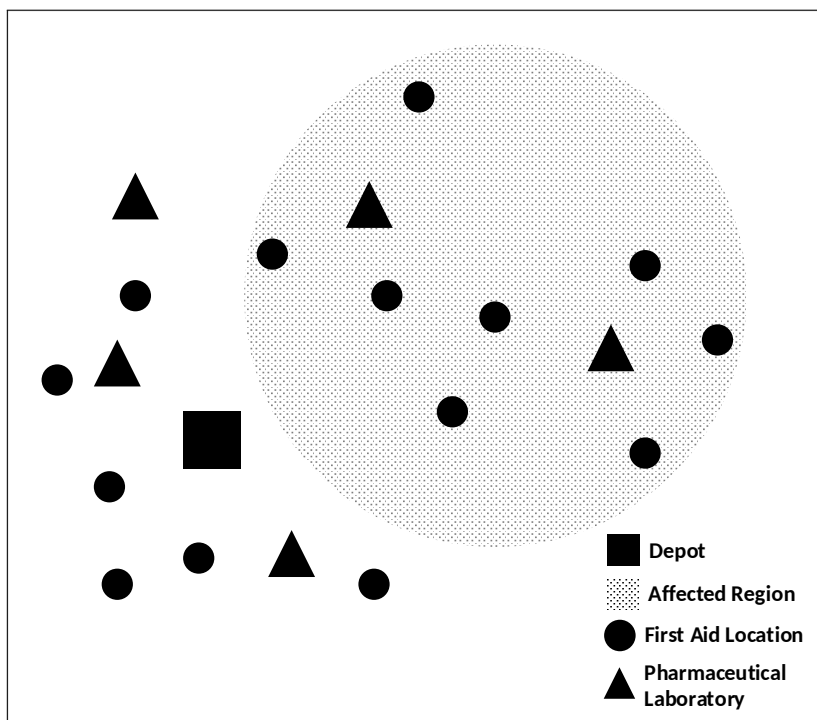


Figure 4.1: An example of a situation picture provided by the sensors.

The drones are available and initially located at the depot. The capacity of drones is utilized to perform the required delivery of raw material from the depot to intermediate points and then to final points, in order to minimize the total transport time and guarantee the replenishment of medical supplies. The solution to this problem is given by a set of drone routes. Each drone starts from the depot, visits a set of PLs and FALs, and returns empty to the depot. The goal is to minimize the total duration of the delivery routes such that:

1. Every route must start and end at the depot.
2. The routes do not exceed the maximum tour length.
3. Every PL or FAL is visited by only one drone and only once.
4. The total delivery demand from the depot to the PLs does not exceed the drone capacity.
5. The total demand of a FAL to be served from the PLs with a certain drug cannot exceed the available inventory of this drug at the PLs.
6. The PLs designated to satisfy the demand of a FAL must be visited before the point and by the same delivery drone.
7. Decisions on distribution plans need to be made in real-time.

Notice that, differently from the previous introduction of the OCVRP, the consideration of “decisions on distribution plans need to be made in real-time” has been added to the problem.

The distribution network of this problem can be defined as a directed graph $G = (V, A)$. The set V is composed of the central depot (node 0), the set of PLs, and the set of FALs. The PLs and FALs are served by the same fleet of homogeneous drones with a certain capacity, which is available at the depot at the start time t_0 . The complete mathematical formulation of this problem can be found in Abdulkader et al. (2018) and Bayliss et al. (2020b). Figure 4.2 provides a route example for the disaster scenario presented in Figure 4.1. In this example, PL 2 serves FAL 7, while PL 3 serves FALs 9 and 6. It might be the case that FALs 9 or 6 are served by PL 2. In this case, the same visit order would be preserved as the pick-up would be performed prior to delivery. In the second route, PL 1 serves FALs 4, 5, 10, 8, and 11. The routes satisfy constraints on vehicle capacity, maximum tour length, and inventory availability.

4.1.3 Solution Method: an Agile Optimization Framework

Agile optimization has arisen as a new optimization concept for real-time decision making. As introduced, it refers to the massive parallelization of BR algorithms, which are extremely fast in execution, easily parallelizable, flexible, and require the fine-tuning of few, or even just a single parameter. The idea behind this technique is to run in parallel several hundred or even thousands of threads, each thread being an execution of a BR heuristic. As a result, several alternative solutions are generated in the same wall-clock time as the one employed by the original heuristic, i.e., milliseconds in most cases. Then, different solutions are provided –some of them outperforming the one generated by the original heuristic– and the best solution is chosen. By using skewed probability distributions, BR techniques employ the idea of introducing a biased (non-symmetric) randomization effect into a heuristic procedure. As a result, a deterministic heuristic –which is extremely fast in execution, even for

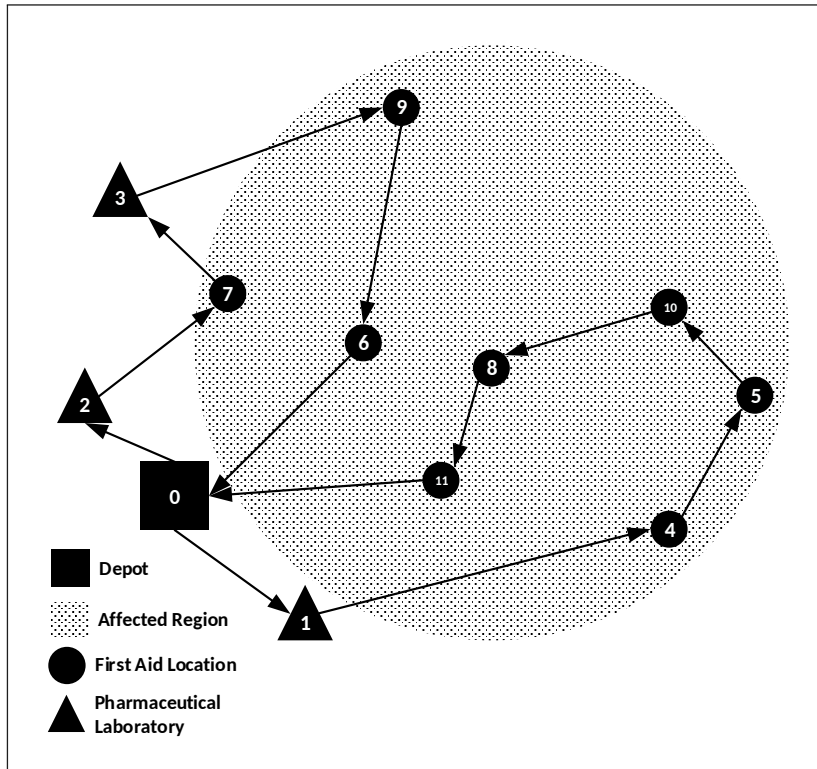


Figure 4.2: An example of a 2-echelon distribution system.

large-scale optimization problems—is extended into a probabilistic algorithm without losing the logic behind the original heuristic.

Agile optimization represents a new optimization perspective which allows to find reasonably good solutions for large-scale and *NP-hard* optimization problems in real-time. This concept is also necessary when dealing with dynamic systems (e.g., traffic, vehicles location, unexpected demands, disruptions, etc.), where the environmental conditions are continuously changing and re-optimization of the system is required every few minutes or even seconds.

To solve this OCVRP in the context of humanitarian logistics, the *BRLH* heuristic, which is composed of a biased-randomized constructive stage plus a local search procedure (introduced in Section 3.1.3.3), is embedded in a parallel framework to complete the agile optimization approach. Therefore, multiple runs of the same instance are executed in a concurrent / parallel way using different seeds for the pseudo-random number generator. This is possible due to the non-existence of dependencies among the different runs. Hence, several solutions are generated at the same time, and the one with the lowest cost is chosen.

Pseudocode 12 presents the structure of the *AO* approach. It is assumed that the number of processors available for executing the *AO* is greater or equal to the number of desired threads. Otherwise, an overhead can destroy the basic principle of this strategy. After defining the number of parallel runs to be performed, n_{th} , the respective executions of the *BRLH* are called (line 3), being each one fed with a β value, which can be different and, therefore, generates different solutions according to the variability of this parameter. Each solution is inserted into a pool of solutions (line 4). Once the solutions are computed, the best-found

solution is sought (line 6) returned by the approach (line 7).

Pseudocode 12: Agile Optimization Framework

Data: set of nodes V , geometric distribution parameter β , number of threads n_{th}

```

1 Function AO( $V, \beta, n_{th}$ ):
2    $solutions \leftarrow \{\}$ 
3   forall  $n_{th}$  available processors do
4      $current_{sol} \leftarrow BRLH(V, \beta)$ 
5      $solutions \leftarrow solutions \cup \{current_{sol}\}$ 
6   end
7    $best_{sol} \leftarrow getBestSolution(solutions)$ 
8   return  $best_{sol}$ 
9 End

```

4.1.4 Computational Experiments and Results

The proposed AO solution approach was tested against the 80 instances proposed by Abdulkader et al. (2018) for the classical OCVRP. As described, these instances differ in the number of retail stores and customers. In the current application context, the retail centers, R , are equivalent to PLs and consumers, C , to the FALs. Similarly to the previous problem application, the maximum tour length and the vehicle capacity are fixed to 8 hours and 100 weight units, respectively, and the parameter β is chosen in the interval $[0.45, 0.75]$. Repeating the same problem parameters setting allows us the proper comparison among different solving approaches. The number of parallel runs for the AO strategy was fixed with 64, $AO_{(64)}$, each run using a different seed for the pseudo-random number generator. For each instance, a total of 10 repetitions were performed in order to collect statistical data. The performance of the heuristics was measured by means of the percentage gap between the best-found solution using that methodology. The algorithm was developed in Java and the tests were performed on an Intel Core i7-8550U processor with 16 GB of RAM.

Table 4.1 presents the results obtained by the agile approach on the set of small-sized instances. Apart from considering the two approaches proposed by Abdulkader et al. (2018) (i.e., AH and MAC), the case in which BRLH employs only a single thread/run for generating a single solution, and the case where the method is allowed to generate solutions during a limited amount of time and then returning the best-found one when this stop criterion is met, are considered. The latter is set to run during a maximum execution time of 5 seconds. These approaches are referred to as BRLH and $BRLH_{(5)}$, respectively. For each instance, the following information is provided: the solution cost obtained by the different methodologies ($BRLH, AO_{(64)}, BRLH_{(5)}, AH, MAC$), the average cost and percentage standard deviation (% SD) of our results, the CPU time (in seconds) required by each methodology, and their gaps. Since BRLH differs from $AO_{(64)}$ only in the number of parallel runs, their CPU time is aggregated in a single column. According to Abdulkader et al. (2018), the CPU time required by both AH and MAC methods to generate the solutions for the small-sized instances was less than one second. Nevertheless, we have freely implemented the AH heuristic in order to collect the CPU time in our machine environment, which is less powerful than their one.

Table 4.1: Comparison of results obtained by the different methodologies on solving small-sized instances.

<i>I</i>	<i> R </i>	<i> C </i>						<i>Avg. Cost</i> (2)	<i>% SD</i> (2)	<i>Time (sec.)</i>			<i>gap</i>					
			<i>1</i> <i>BRLH</i>	<i>2</i> <i>AO₍₆₄₎</i>	<i>3</i> <i>BRLH₍₅₎</i>	<i>4</i> <i>AH</i>	<i>5</i> <i>MAC</i>			(1, 2)	(4)	(5)	(4-1)	(4-2)	(4-3)	(5-1)	(5-2)	(5-3)
a1	3	6	398.3	386.9	386.9	479.2	386.9*	386.9	0.0%	0.0	0.0	<1	-17%	-19%	-19%	3%	0%	0%
a2	3	9	430.2	416.3	416.3	589.6	416.4*	416.3	0.0%	0.0	0.0	<1	-27%	-29%	-29%	3%	0%	0%
a3	3	12	476.2	438.3	427.7	512.3	424.3*	447.9	2.2%	0.0	0.0	<1	-7%	-14%	-17%	12%	3%	1%
a4	3	15	571.7	500.6	487.7	767.1	455.3*	513.0	2.4%	0.0	0.0	<1	-25%	-35%	-36%	26%	10%	7%
a5	3	18	714.3	675.6	661.0	936.2	601.4*	690.4	1.3%	0.0	0.0	<1	-24%	-28%	-29%	19%	12%	10%
a6	4	6	454.3	425.7	425.0	512.1	419.3*	435.8	2.1%	0.0	0.0	<1	-11%	-17%	-17%	8%	2%	1%
a7	4	9	463.5	455.9	455.9	688.4	455.9*	462.0	0.7%	0.0	0.0	<1	-33%	-34%	-34%	2%	0%	0%
a8	4	12	483.0	479.7	466.3	711.9	449.4	484.1	1.0%	0.0	0.0	<1	-32%	-33%	-34%	7%	7%	4%
a9	4	15	530.5	477.6	461.4	746.6	457.3*	518.5	4.7%	0.0	0.0	<1	-29%	-36%	-38%	16%	4%	1%
a10	4	18	718.9	608.5	602.8	740.8	514.4*	676.6	5.3%	0.0	0.0	<1	-3%	-18%	-19%	40%	18%	17%
a11	5	6	512.1	492.0	488.5	545.7	486.0*	495.7	1.1%	0.0	0.0	<1	-6%	-10%	-10%	5%	1%	1%
a12	5	9	688.9	624.8	624.8	882.0	624.8*	637.3	3.3%	0.0	0.0	<1	-22%	-29%	-29%	10%	0%	0%
a13	5	12	626.7	579.5	539.0	961.7	535.4*	618.6	3.0%	0.0	0.0	<1	-35%	-40%	-44%	17%	8%	1%
a14	5	15	739.1	680.2	656.8	838.9	605.0*	710.3	2.7%	0.0	0.0	<1	-12%	-19%	-22%	22%	12%	9%
a15	5	18	850.6	777.9	747.4	898.8	709.9	804.6	2.4%	0.0	0.0	<1	-5%	-13%	-17%	20%	10%	5%
a16	6	6	505.5	502.6	484.5	582.1	468.9*	505.0	0.2%	0.0	0.0	<1	-13%	-14%	-17%	8%	7%	3%
a17	6	9	500.3	481.0	468.6	609.0	468.6*	491.0	2.0%	0.0	0.0	<1	-18%	-21%	-23%	7%	3%	0%
a18	6	12	715.2	659.3	638.8	924.4	586.6*	670.1	1.2%	0.0	0.0	<1	-23%	-29%	-31%	22%	12%	9%
a19	6	15	796.9	767.2	751.5	964.1	750.9*	781.3	1.6%	0.0	0.0	<1	-17%	-20%	-22%	6%	2%	0%
a20	6	18	778.8	717.6	674.6	1,005.7	601.7	743.3	3.3%	0.0	0.1	<1	-23%	-29%	-33%	29%	19%	12%
<i>Average</i>								2.0%	0.0	0.0	0.0	-	-19%	-24%	-26%	14%	7%	4%

(*) Optimal solutions provided by Abdulkader et al. (2018)

When comparing the $AO_{(64)}$ solution method with those proposed by Abdulkader et al. (2018), we are able to find results, on average, 24% better than the AH , column *gap* (4-2), and 7% worse than the MAC , column *gap* (5-2). In terms of CPU time, our results are quite competitive mainly when compared with the ones from the AH method, which require the same computational time but are significantly worse. Although our method requires basically the same CPU time as MAC for solving these small instances, this is not the case when larger-sized instances are considered, in which MAC time is orders of magnitude larger than the one requested by our $AO_{(64)}$. Comparing with the optimal solutions provided by Abdulkader et al. (2018), our methodology is able to find four optimal solutions. By allowing 5 seconds during the execution of our methodology, the resulting $BRLH_{(5)}$ is able to find six optimal solutions and five near-optimal solutions (solutions up to 1% worse than the optimal ones), column *gap* (5-3), being only 4% worse than the MAC results, and 26% better than the AH results, column *gap* (4-3). Analyzing the single thread version of our $BRLH$, its results are, on average, 14% worse than MAC , column *gap* (5-1), but approximately 19% better than the alternative AH heuristic, column *gap* (4-1). Regarding the variance of our results, the average standard deviation is only 2%.

Another characteristic which directly influences the performance of our methodology is the number of threads, i.e., its number of parallel runs. Therefore, in order to verify the behavior on different hardware settings which are characterized by a different number of threads, our AO was tested on settings consisting of 1, 8, 16, 32, 64, 128, 256, 512, 1,024, 2,048, and 4,096 threads. Figure 4.3 presents the convergence of the solutions when compared with the MAC solutions. By increasing the number of threads, our methodology is able to find up to 9 optimal solutions when using more than 1,024 threads, achieving a positive average gap of 3.6%.

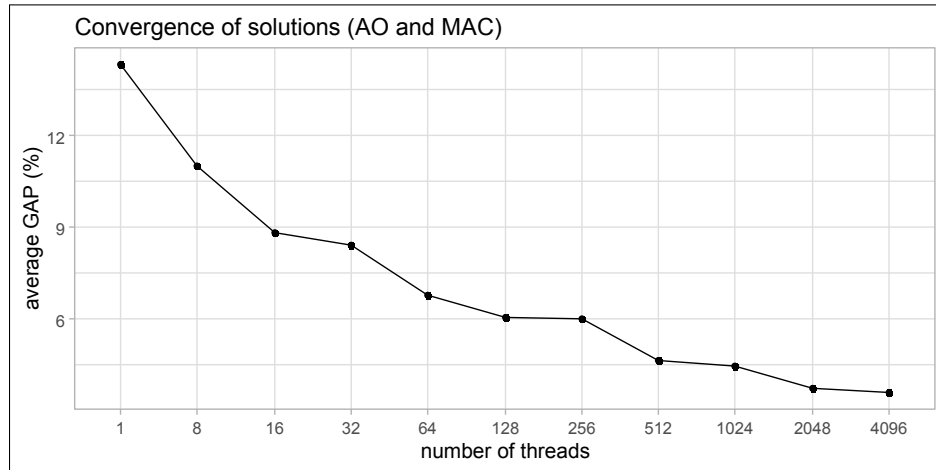


Figure 4.3: Comparison between *BRLH* and *MAC* on solving small-sized instances with a different number of parallel runs.

Due to the satisfactory performance of the *AO* approach in small-sized instances, it was also tested in solving large-sized instances. Although the deterministic version of the heuristic (*LH*) has been introduced and discussed in the previous chapter, it is considered in the following analyses to study the benefits of its extension into an agile approach. Tables 4.2, 4.3 and 4.4 present the obtained results on scenarios of tight, relaxed and abundant inventory, respectively.

The first analysis aims to quantify the improvement that is achieved when extending the deterministic version of our proposed methodology into a biased-randomized algorithm and agile optimization strategy. As we can see in column *gap* (1-2) of Tables 4.2, 4.3 and 4.4, which compares the deterministic heuristic with its biased-randomized version, the *BRLH* is able to improve up to 6% of the results by only introducing a biased-random behavior at the constructive algorithm. By analysing column *gap* (1-3), with the use of agile optimization, the resulting heuristic is able to improve its deterministic version at between 8% and 11%, without increasing the required CPU time. This particularity of our methodology, which refers to parallel executions of BR algorithms, allow us to generate multiple alternative solutions in the same clock time. By allowing the $BRLH_{(5)}$ to run for five seconds, column *gap* (1-4), the results achieve a negative gap from 11% to 14%. It highlights the potential of this solving approach, however, due to the application context, the time required to generate a feasible solution is a hard-constraint, therefore, limiting time consuming executions of this methodology. Comparing the results generated by our *AO* with those generated by the *AH* methodology, column *gap* (5-3), our approach can improve at least 19% of the solutions on abundant inventory scenario and between 24% and 25% on the remaining ones. Regarding the $BRLH_{(5)}$, column *gap* (5-4), an average improvement of about 3% is achieved with respect the *AO*. Now, comparing the *AO* results with the best-known solutions, column *gap* (6-3), our results are at most 18% worst on average (considering all the solutions). However, the average CPU time required by the *MAC* is hugely more extensive than ours, which does not represent a suitable time for our application context that requires solutions in extremely short computational times. Note that for some instances, *MAC* is able to generate

Table 4.2: Comparison of results obtained by the different methodologies in the scenario of tight inventory.

I	R	C	Time (sec.)						gap														
			1 LH	2 BRLH	3 AO ₍₆₄₎	4 BRLH ₍₅₎	5 AH	6 MAC	(1-2)	(1-3)	(1-4)	(5-2)	(5-3)	(5-4)	(6-2)	(6-3)	(6-4)						
b1	10	25	1,277.5	1,199.8	1,134.5	1,095.1	1,631.6	1,002.5	1,153.0	1.7%	0.0	1.1	0.1	7.0	-6%	-11%	-14%	-26%	-30%	-33%	20%	13%	9%
b2	10	50	1,641.1	1,495.4	1,392.5	1,345.9	2,057.5	1,192.0	1,448.1	1.5%	0.0	0.2	0.0	47.0	-9%	-15%	-18%	-27%	-32%	-35%	25%	17%	13%
b3	10	75	2,663.7	2,678.5	2,463.4	2,264.7	3,006.2	1,815.4	2,554.8	1.9%	0.0	1.9	0.0	79.0	1%	-8%	-15%	-11%	-18%	-25%	48%	36%	25%
b4	10	100	2,415.1	2,189.5	1,968.7	1,865.2	2,830.2	1,529.0	2,062.3	2.5%	0.0	3.7	0.0	286.0	-9%	-18%	-23%	-23%	-30%	-34%	43%	29%	22%
b5	10	150	2,678.2	2,524.8	2,395.0	2,327.8	3,478.7	1,905.2	2,450.4	1.3%	0.0	4.9	0.0	576.0	-6%	-11%	-13%	-27%	-31%	-33%	33%	26%	22%
b6	15	25	1,540.5	1,474.4	1,408.1	1,379.1	1,774.4	1,313.7	1,428.9	1.0%	0.0	1.7	0.0	7.0	-4%	-9%	-10%	-17%	-21%	-22%	12%	7%	5%
b7	15	50	2,059.0	1,979.9	1,788.8	1,680.6	2,461.8	1,522.3	1,898.4	3.0%	0.0	2.1	0.0	44.0	-4%	-13%	-18%	-20%	-27%	-32%	30%	18%	10%
b8	15	75	3,105.3	2,757.7	2,649.4	2,521.1	3,545.1	2,101.8	2,678.4	0.7%	0.0	3.4	0.0	131.0	-11%	-15%	-19%	-22%	-25%	-29%	31%	26%	20%
b9	15	100	3,121.5	2,964.5	2,826.3	2,781.9	3,529.0	2,329.5	2,905.7	1.5%	0.0	1.4	0.0	209.0	-5%	-9%	-11%	-16%	-20%	-21%	27%	21%	19%
b10	15	150	4,292.4	3,953.0	3,749.6	3,707.5	4,916.8	3,012.2	3,858.2	1.3%	0.1	4.7	0.0	430.0	-8%	-13%	-14%	-20%	-24%	-25%	31%	24%	23%
b11	20	25	2,035.2	1,920.4	1,889.5	1,818.1	2,432.6	1,611.3	1,914.0	0.7%	0.0	3.6	0.0	11.0	-6%	-7%	-11%	-21%	-22%	-25%	19%	17%	13%
b12	20	50	2,335.4	2,187.8	2,126.0	2,068.8	2,695.3	1,800.9	2,160.0	0.8%	0.0	1.1	0.0	50.0	-6%	-9%	-11%	-19%	-21%	-23%	21%	18%	15%
b13	20	75	3,212.7	2,973.1	2,784.6	2,689.8	3,936.7	2,406.0	2,913.4	1.9%	0.0	0.7	0.0	127.0	-7%	-13%	-16%	-24%	-29%	-32%	24%	16%	12%
b14	20	100	3,025.2	3,070.4	2,869.4	2,800.3	3,826.1	2,483.8	2,933.1	1.1%	0.0	2.7	0.0	327.0	1%	-5%	-7%	-20%	-25%	-27%	24%	16%	13%
b15	20	150	3,934.3	3,307.2	3,241.8	3,226.5	4,496.1	2,679.2	3,345.4	2.1%	0.1	4.9	0.0	708.0	-16%	-18%	-18%	-26%	-28%	-28%	23%	21%	20%
b16	25	25	2,019.4	1,950.8	1,897.7	1,824.8	2,254.9	1,669.6	1,925.4	1.3%	0.0	4.2	0.0	13.0	-3%	-6%	-10%	-13%	-16%	-19%	17%	14%	9%
b17	25	50	2,665.9	2,523.3	2,430.3	2,370.4	3,020.8	1,965.6	2,494.4	1.5%	0.0	2.1	0.0	46.0	-5%	-9%	-11%	-16%	-20%	-22%	28%	24%	21%
b18	25	75	3,207.6	3,044.9	2,946.4	2,803.8	3,963.5	2,449.8	2,974.5	0.7%	0.0	2.1	0.0	136.0	-5%	-8%	-13%	-23%	-26%	-29%	24%	20%	14%
b19	25	100	4,064.0	3,730.2	3,502.3	3,478.3	4,933.9	2,788.5	3,622.3	1.9%	0.1	3.7	0.0	257.0	-8%	-14%	-14%	-24%	-29%	-30%	34%	26%	25%
b20	25	150	3,782.7	3,707.4	3,544.4	3,405.8	4,721.3	2,890.3	3,593.0	1.1%	0.2	3.4	0.0	712.0	-2%	-6%	-10%	-21%	-25%	-28%	28%	23%	18%
<i>Average</i>									1.5%						-6%								
									0.0						2.7								
									0.0						210.2								
									-25%						-28%								
									-21%						27%								
									21%						16%								

noticeable better solutions in reduced computing times (less than a minute). However, this is only the case for small instances. In the abundant inventory scenario, our methodology is

Table 4.3: Comparison of results obtained by the different methodologies in the scenario of relaxed inventory.

<i>I</i>	<i> R </i>	<i> C </i>	Time (sec.)						gap														
			1 <i>LH</i>	2 <i>BRLH</i>	3 <i>AO₍₆₄₎</i>	4 <i>BRLH₍₅₎</i>	5 <i>AH</i>	6 <i>MAC</i>	(1-2)	(1-3)	(1-4)	(5-2)	(5-3)	(5-4)	(6-2)	(6-3)	(6-4)						
b21	10	25	1,233.0	1,220.6	1,087.6	1,007.1	1,571.6	879.2	1,110.3	1.5%	0.0	3.8	0.0	10.0	-1%	-12%	-18%	-22%	-31%	-36%	39%	24%	15%
b22	10	50	1,490.8	1,406.4	1,355.9	1,197.5	1,920.6	1,083.7	1,374.4	1.0%	0.0	0.0	0.0	85.0	-6%	-9%	-20%	-27%	-29%	-38%	30%	25%	11%
b23	10	75	2,468.0	2,190.6	2,121.2	2,008.2	2,699.2	1,591.5	2,156.9	1.0%	0.0	2.2	0.0	167.0	-11%	-14%	-19%	-19%	-21%	-26%	38%	33%	26%
b24	10	100	1,885.0	1,761.5	1,645.9	1,599.2	2,305.1	1,437.7	1,706.0	2.3%	0.0	2.2	0.0	528.0	-7%	-13%	-15%	-24%	-29%	-31%	23%	14%	11%
b25	10	150	1,998.6	2,136.9	2,003.1	1,875.2	2,700.4	1,520.5	2,030.3	0.8%	0.0	3.5	0.0	1,836.0	7%	0%	-6%	-21%	-26%	-31%	41%	32%	23%
b26	15	25	1,591.4	1,462.0	1,268.0	1,268.0	1,665.2	1,180.8	1,353.0	3.5%	0.0	0.6	0.0	11.0	-8%	-20%	-20%	-12%	-24%	-24%	24%	7%	7%
b27	15	50	1,940.7	1,756.4	1,674.9	1,603.5	2,320.7	1,329.3	1,692.0	0.8%	0.0	3.2	0.0	73.0	-9%	-14%	-17%	-24%	-28%	-31%	32%	26%	21%
b28	15	75	2,436.3	2,269.4	2,204.4	2,076.9	3,016.5	1,692.4	2,255.4	1.3%	0.0	2.0	0.0	279.0	-7%	-10%	-15%	-25%	-27%	-31%	34%	30%	23%
b29	15	100	2,648.3	2,628.0	2,430.1	2,299.9	3,302.4	2,016.4	2,486.5	1.7%	0.0	1.2	0.0	567.0	-1%	-8%	-13%	-20%	-26%	-30%	30%	21%	14%
b30	15	150	3,373.2	2,963.9	2,763.6	2,767.0	3,919.0	2,399.6	2,890.3	2.6%	0.1	0.5	0.0	1,407.0	-12%	-18%	-18%	-24%	-29%	-29%	24%	15%	15%
b31	20	25	1,835.5	1,774.4	1,687.2	1,645.5	1,993.5	1,495.8	1,747.6	1.6%	0.0	2.4	0.0	16.0	-3%	-8%	-10%	-11%	-15%	-17%	19%	13%	10%
b32	20	50	2,320.5	2,140.7	2,005.0	1,867.8	2,713.0	1,656.9	2,045.6	1.1%	0.0	3.9	0.0	76.0	-8%	-14%	-20%	-21%	-26%	-31%	29%	21%	13%
b33	20	75	2,404.6	2,360.4	2,312.0	2,232.3	3,393.3	1,799.6	2,348.2	0.7%	0.0	1.5	0.0	262.0	-2%	-4%	-7%	-30%	-32%	-34%	31%	28%	24%
b34	20	100	2,751.9	2,677.7	2,497.7	2,423.8	3,127.5	2,018.5	2,532.6	1.0%	0.0	2.5	0.0	740.0	-3%	-9%	-12%	-14%	-20%	-23%	33%	24%	20%
b35	20	150	3,157.4	2,832.5	2,818.7	2,739.6	3,742.2	2,291.0	2,877.0	1.3%	0.1	0.6	0.0	2,141.0	-10%	-11%	-13%	-24%	-25%	-27%	24%	23%	20%
b36	25	25	1,844.0	1,786.4	1,744.7	1,643.7	2,032.1	1,550.0	1,765.6	0.8%	0.0	0.5	0.0	15.0	-3%	-5%	-11%	-12%	-14%	-19%	15%	13%	6%
b37	25	50	2,663.9	2,555.2	2,409.1	2,279.3	3,130.5	1,939.5	2,455.3	1.1%	0.0	0.4	0.0	73.0	-4%	-10%	-14%	-18%	-23%	-27%	32%	24%	18%
b38	25	75	2,790.7	2,770.3	2,547.4	2,507.6	3,433.2	2,088.6	2,608.6	1.1%	0.0	3.5	0.0	283.0	-1%	-9%	-10%	-19%	-26%	-27%	33%	22%	20%
b39	25	100	3,352.9	3,207.2	3,076.8	2,964.6	3,824.5	2,244.1	3,113.1	1.0%	0.0	4.5	0.0	656.0	-4%	-8%	-12%	-16%	-20%	-22%	43%	37%	32%
b40	25	150	2,971.1	2,926.0	2,811.1	2,764.8	3,447.9	2,229.4	2,848.8	0.9%	0.1	4.5	0.0	2,077.0	-2%	-5%	-7%	-15%	-18%	-20%	31%	26%	24%
<i>Average</i>									1.3%	0.0	2.2	0.0	565.1	-5%	-10%	-14%	-20%	-24%	-28%	30%	23%	18%	

only 9% worse, being able to find a new best solution for instance *b42*. When comparing the *BRLH* results, its performance is, on average, 24% worse than *MAC*, column *gap* (6-2), but

Table 4.4: Comparison of results obtained by the different methodologies in the scenario of abundant inventory.

I	$ R $	$ C $	Time (sec.)						gap													
			1 LH	2 BRLH	3 AO ₍₆₄₎	4 BRLH ₍₅₎	5 AH	6 MAC	(1-2)	(1-3)	(1-4)	(5-2)	(5-3)	(5-4)	(6-2)	(6-3)	(6-4)					
			Avg. Cost			% SD																
			(3)	(3)	(3)	(3)	(3)	(3)	(2,3)	(4)	(5)	(6)	(1-2)	(1-3)	(1-4)	(5-2)	(5-3)	(5-4)	(6-2)	(6-3)	(6-4)	
b41	10	25	805.4	793.4	760.8	743.9	897.6	711.3	775.9	1.5%	0.0	0.0	16.0	-1%	-6%	-8%	-12%	-15%	-17%	12%	7%	5%
b42	10	50	1,014.2	910.4	872.9	870.0	1,287.8	875.2	885.0	1.2%	0.0	4.6	143.0	-10%	-14%	-14%	-29%	-32%	-32%	4%	0%	-1%
b43	10	75	1,463.5	1,332.3	1,260.3	1,225.6	1,531.1	1,132.1	1,307.0	1.7%	0.0	4.1	358.0	-9%	-14%	-16%	-13%	-18%	-20%	18%	11%	8%
b44	10	100	1,379.4	1,382.4	1,303.3	1,280.8	1,636.5	1,224.1	1,309.9	0.5%	0.0	4.8	978.0	0%	-6%	-7%	-16%	-20%	-22%	13%	6%	5%
b45	10	150	1,499.3	1,509.9	1,415.9	1,353.9	1,551.8	1,273.9	1,438.0	1.1%	0.0	4.0	2,085.0	1%	-6%	-10%	-3%	-9%	-13%	19%	11%	6%
b46	15	25	1,137.5	1,063.2	1,039.6	1,013.5	1,264.3	996.9	1,048.0	0.7%	0.0	0.2	22.0	-7%	-9%	-11%	-16%	-18%	-20%	7%	4%	2%
b47	15	50	1,247.6	1,225.7	1,154.2	1,118.8	1,488.1	1,080.3	1,186.0	1.1%	0.0	1.9	159.0	-2%	-7%	-10%	-18%	-22%	-25%	13%	7%	4%
b48	15	75	1,595.3	1,483.4	1,370.0	1,349.6	1,815.2	1,252.4	1,401.3	2.6%	0.0	4.8	559.0	-7%	-14%	-15%	-18%	-25%	-26%	18%	9%	8%
b49	15	100	2,021.3	1,979.9	1,785.9	1,748.2	2,242.4	1,594.0	1,867.7	1.9%	0.0	4.4	1,167.0	-2%	-12%	-14%	-12%	-20%	-22%	24%	12%	10%
b50	15	150	2,059.6	1,941.5	1,869.6	1,841.3	2,459.5	1,691.4	1,901.2	0.7%	0.1	1.4	4,126.0	-6%	-9%	-11%	-21%	-24%	-25%	15%	11%	9%
b51	20	25	1,507.2	1,476.0	1,425.7	1,350.3	1,660.9	1,302.9	1,446.8	0.7%	0.0	1.2	33.0	-2%	-5%	-10%	-11%	-14%	-19%	13%	9%	4%
b52	20	50	1,464.7	1,392.5	1,367.8	1,357.2	1,740.7	1,301.0	1,379.3	0.4%	0.0	3.8	156.0	-5%	-7%	-7%	-20%	-21%	-22%	7%	5%	4%
b53	20	75	1,797.7	1,757.5	1,626.2	1,559.9	2,096.8	1,421.8	1,661.6	1.0%	0.0	2.5	605.0	-2%	-10%	-13%	-16%	-22%	-26%	24%	14%	10%
b54	20	100	2,066.2	2,084.0	1,839.8	1,801.0	2,226.4	1,640.6	1,956.5	2.6%	0.0	3.1	1,370.0	1%	-11%	-13%	-6%	-17%	-19%	27%	12%	10%
b55	20	150	2,214.0	2,073.1	2,044.3	1,963.2	2,518.2	1,763.3	2,072.3	0.8%	0.1	1.3	5,321.0	-6%	-8%	-11%	-18%	-19%	-22%	18%	16%	11%
b56	25	25	1,423.2	1,399.9	1,382.4	1,356.0	1,550.7	1,311.6	1,395.9	0.5%	0.0	1.2	36.0	-2%	-3%	-5%	-10%	-11%	-13%	7%	5%	3%
b57	25	50	1,670.8	1,631.2	1,548.5	1,532.8	1,835.4	1,468.1	1,599.4	1.4%	0.0	4.6	203.0	-2%	-7%	-8%	-11%	-16%	-16%	11%	5%	4%
b58	25	75	2,047.5	1,922.6	1,870.0	1,776.1	2,276.9	1,654.9	1,899.1	1.0%	0.0	1.5	791.0	-6%	-9%	-13%	-16%	-18%	-22%	16%	13%	7%
b59	25	100	1,856.4	1,860.7	1,808.1	1,743.5	2,061.9	1,575.7	1,829.8	0.7%	0.0	3.1	1,262.0	0%	-3%	-6%	-10%	-12%	-15%	18%	15%	11%
b60	25	150	1,968.1	1,940.8	1,856.1	1,820.7	2,347.8	1,653.3	1,878.0	0.8%	0.1	2.1	4,549.0	-1%	-6%	-7%	-17%	-21%	-22%	17%	12%	10%
Average									1.2%	0.0	2.7	0.0	1,197.0	-3%	-8%	-11%	-15%	-19%	-21%	15%	9%	6%

approximately 19% better, on average, than the AH heuristic, column gap (5-2). Regarding the BRLH₍₅₎, an average improvement of about 5% is achieved, when comparing against

MAC, column *gap* (6-4), with respect the AO, column *gap* (6-3). Regarding the variance of our results, the average percentage standard deviation varies from 1.2% to 1.5%. Additionally, all the average values are below the solution cost obtained by the AH heuristic. These last two analyses allow us to certify the robustness of our proposed methodology, which is capable of generating good solutions with a small variance of the cost of the solutions.

Similar to the set of small-sized instances, Figures 4.4 and 4.5 present the convergence of the large-sized instances solutions, generated by the AO, when comparing with both the AH and MAC methodologies, for each inventory scenario.

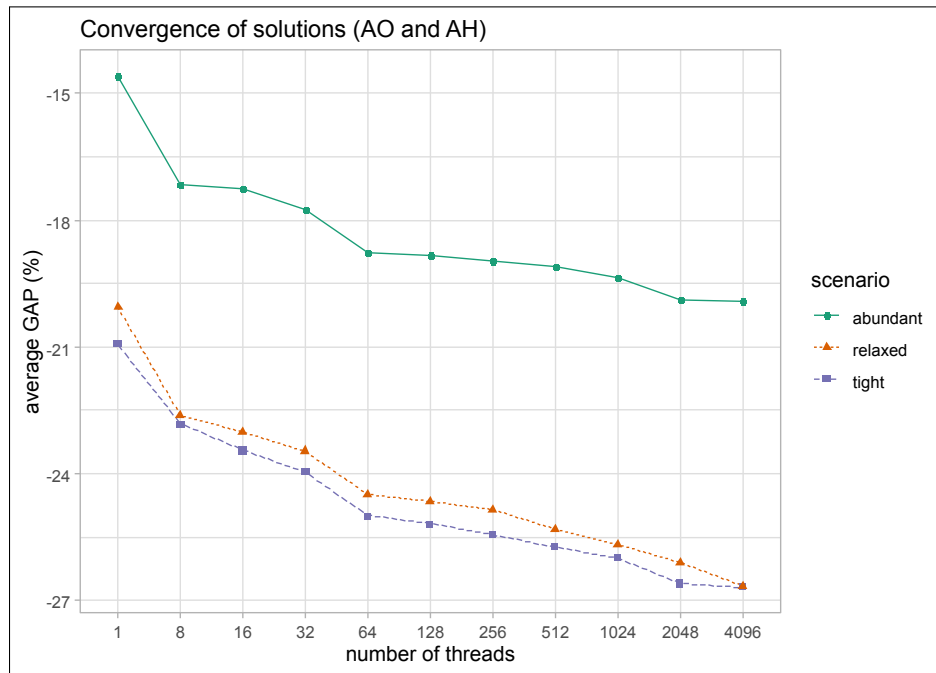


Figure 4.4: Comparison between AO and AH on solving large-sized instances with a different number of parallel runs.

As we can see, the convergence behavior is similar for all inventory scenarios when comparing with AH and MAC approaches. By increasing the number of threads, our methodology is able to reach up to 27% of improvement when comparing with AH, being only 8% worse than MAC's results. However, it must be highlighted that increasing the number of parallel runs is restricted to the hardware capacities of the machine. In other words, the lack of resources can destroy the idea behind agile optimization, in which the performance -in terms of execution time- is degraded as the number of threads increases, due to changes of context and the use of the resources.

4.1.5 Conclusions

In this chapter, an agile optimization approach was proposed to solve a two-echelon vehicle routing problem. In our case, this problem is motivated by the distribution of drugs with drones in disaster situations, where the affected areas might become no longer accessible by conventional cargo vehicles. This might be the case in rescue operations in humanitarian logistics, where every second can be decisive to save lives. Therefore, our methodology

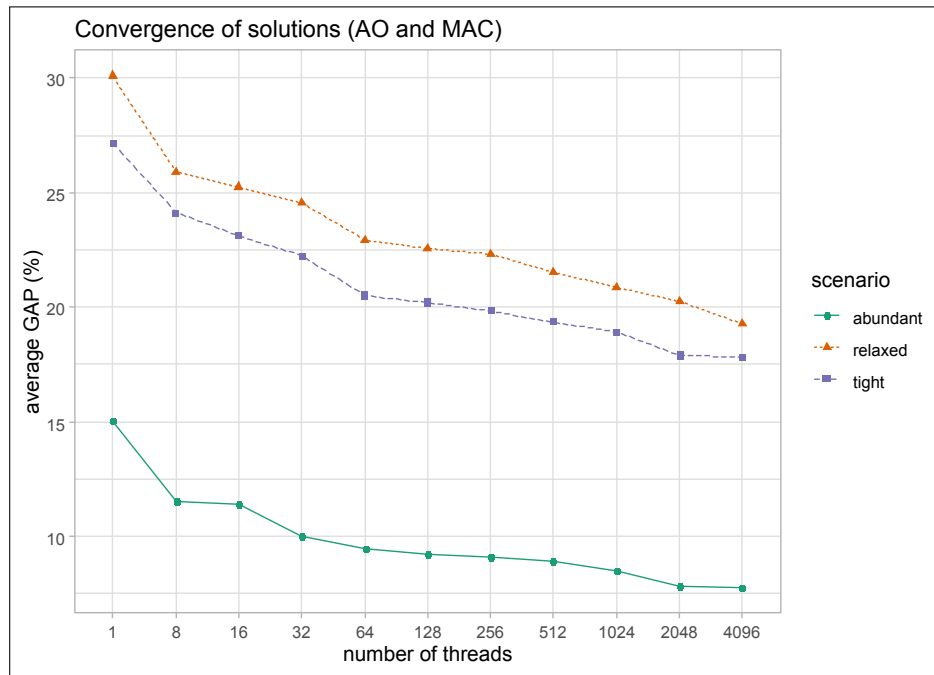


Figure 4.5: Comparison between AO and MAC on solving large-sized instances with a different number of parallel runs.

must be able to provide good solutions in a very short computational time. The concept of agile optimization was introduced in order to meet this goal.

As results show, the use of BR techniques together with parallel computing is able to improve about 10% the solutions of the deterministic version of our heuristic, without increasing the required wall-clock time. This is an attractive characteristic of the proposed approach, which allows us to take advantage of the current devices, which are more and more efficient nowadays.

In general, our methodology showed to be very competitive when solving both small- and large-sized instances. At this point, we contrast with the alternative solution methods which are efficient but require high computational times to provide high-quality solutions. In both set of instances, small- and large-sized, we were able to find results in real-time (within milliseconds) that were quite close to the best-known solutions. Particularly, in the abundant inventory scenario of large-sized instances, our methodology was able to generate a new better solution than the best-known one. Increasing the number of parallel threads, the performance of our BR heuristic is even better. However, it is a limitation that depends on hardware settings.

Future works include reacting to unexpected events during the planning of the routes, such as to include new emergency locations, to consider the unavailability of stock caused, for instance, by the expansion of the disaster, etc. However, changes in the problem formulation are required in order to deal with this dynamic information and to propose an efficient solution methodology. When incorporating new information, our methodology becomes capable of dealing with the world dynamism, which is continuously changing and being affected by external circumstances and events.

Chapter 5

Applications in Retailing Industry

This chapter ¹ studies the Multi-Period Product Display Problem with Dynamic Attractiveness (MPPDPDA). The problem of selecting, over a multi-period horizon, the most attractive configuration of products to be displayed in a limited space (e.g., a physical table at the store, an advertising brochure, or a website front page), is related to well-known problems such as the product recommendation problem (Adomavicius and Tuzhilin, 2005), the shelf space allocation problem (Hübner and Kuhn, 2012), and the assortment problem (Pentico, 2008). To solve the MPPDPDA, the following solution methods are proposed: (i) a greedy heuristic; (ii) a GRASP metaheuristic; (iii) an ILS metaheuristic; and (iv) a biased-randomized version of each metaheuristic algorithms. A set of new instances has been generated to test both the approaches and compare their performance.

5.1 The Multi-Period Product Display Problem with Dynamic Attractiveness

With the growing rise of internet-based technologies and mobile devices in the last decades, different shopping channels have appeared, emerged, and, consequently, attracted customers' attention. E-commerce is one of these emerging shopping channels, which not only offers customers the possibility of browsing through different stores in an online environment, but also the ability to get a vast of information, opinions, and availability of stock. Thanks to these advances in telecommunication technologies, customers today are changing how they decide where, how, and even when to buy (Verhoef et al., 2017). Moreover, customers are experiencing a unified experience across different shopping platforms, such as a personal computer, a physical retail center, or a mobile device. This fully-integrated approach refers to omnichannel commerce.

In an omnichannel environment, retailers at brick-and-mortar stores have to compete with other shopping channels, and, especially, with the 'showrooming' behavior of customers. Showrooming occurs when customers gather information offline –for instance, by

¹The contents of this chapter are based on the following work:

- Marmol, M.; Martins, L. C.; Hatami, S.; Juan, A. A.; Fernandez, V. (2020): [Using biased-randomized algorithms for the multi-period product display problem with dynamic attractiveness](#). *Algorithms*, 13(2), 34.

visiting the brick-and-mortar store to 'touch and feel' the product– but purchase the product online (Mehra et al., 2013). However, even with customers in the position of choosing where and when to buy, most brands still generate a noticeable part of their sales revenue at brick-and-mortar stores, being them, therefore, of extreme importance and relevant role in capturing customers' attention.

One of the strategies employed by brick-and-mortar retailers to engage more customers is to offer them a variety of attractive products over a multi-period time horizon, which typically covers several weeks. As pointed out by Galino and Moreno (2014), in order to achieve this goal, retailers need to decide which combination of products should be displayed at the store so that the combined attractiveness value is maximized. Some authors define attractiveness as the capacity to cause interest and attract another party's attention (Blau, 2017). According to Ellegaard and Ritter (2007), value creation, interaction process, and emotions define the perceived attractiveness of one actor to another actor. While value creation refers to the potential value, the interaction process deals with trust and commitment. Finally, emotions are the irrational part of decision making, which is not accessible by rational arguments. In this way, attractiveness can be seen as an inter-linked concept that combines value, trust, commitment, and satisfaction (Halinen, 2012). The concept of product attractiveness in retail stores has been also studied by Caro et al. (2014). In their own words: "carrying a static assortment –one that remains the same over time– becomes ineffective and possibly unprofitable because consumers are quickly bored with the choices within the assortment, and they divert their purchases to other consumption options. In other words, the customers' preference for a particular product in the assortment decays over time, as it ages on the shelf." These authors offer several examples of assortment renewal strategies involving clothing retailers, such as H&M and Chico's. They also discuss similar patterns in industries, such as book stores and restaurants, which "frequently change the items on their menu to avoid customer satiation." A similar concept can be found in Bernstein and Albéniz (2017), who claim that "retailers in industries with short life cycles, such as apparel, have started updating their product offering with significant frequency. In particular, fast-fashion retailers, such as Zara or H&M, update their assortments periodically to induce frequent visits to their stores." As exposed in Ferreira and Goh (2019), "assortment rotation has recently been used by both brick-and-mortar and online retailers as a strategy for gaining competitive advantage. A notable category of retailers who have employed this strategy successfully are fast-fashion retailers, such as Zara and H&M, who have differentiated themselves from other retailers by rotating their assortment multiple times throughout the fashion industry standard 6-month selling season."

All in all, the attractiveness value of some products might decay as they are repeatedly displayed. Hence, the retailer must release new products. Similarly, if customers see the same products exposed in a store during several consecutive time periods (days or weeks), their willingness to visit that store will decrease. Accordingly, some popular retailers introduce new products into their stores almost daily. These dependencies across periods are considered in this Section, which constitutes one of the main original contributions of the proposed approach. Schnurr et al. (2017) addresses novelty, placement, and consumers'

opinions as essential factors on the definition of product attractiveness: (i) unfamiliar –or new– products are perceived as more attractive and, consequently, of higher quality when placed in an attractive context; and (ii) the higher the attractiveness score of a product, the higher are the consumers' intentions to purchase it. In this case, given a product included in a display table, its individual attractiveness value is defined as the probability that the product captures the attention (e.g., is selected and analyzed) of a standard customer visiting the shop. Being a probability, it will always be a value between 0 and 1 (or, equivalently, a score between 0 and 100). Attractiveness can also be measured by visual properties, and this is directly related to the existence of a correlation between pairs of products (e.g., products that are complementary or substitute). Thus, retailers have to take into account customers' purchases that occur in channels other than brick-and-mortar stores. Apart from experts' opinion, a large amount of data can be obtained from customers' behavior and preferences in an omnichannel environment. These data can provide retailers with vital information, such as which products generate a higher attraction level among customers of a certain retail store. Hence, identifying the best assortment of products to display has to be made considering customers' preferences (Honhon et al., 2010). Selling strategies for retail stores should have the ability to offer customers a set of different surprising experiences. Using display tables to suggest a set of correlated articles in retail stores is one way to achieve the aforementioned goal. In the apparel sector, for example, a yellow sweater may be positively correlated with a white pair of jeans but negatively correlated with orange trousers (since yellow and orange are not colors that match according to certain fashion trends). Likewise, a skirt might be positively correlated with a top and negatively correlated with a pair of jeans, since both cloth pieces are bottom parts. Again, these dependencies across items should be considered. In practice, data gathered in an omnichannel database could be one of the most efficient ways to determine the correlations between pairs of products.

The problem of selecting, over a multi-period horizon, the most attractive configuration of products to be displayed in a limited space (e.g., a physical table at the store, an advertising brochure, or a website front page), is related to well-known problems such as the product recommendation problem (Adomavicius and Tuzhilin, 2005), the shelf space allocation problem (Hübner and Kuhn, 2012), and the assortment problem (Pentico, 2008). Figure 5.1 shows a simple example to illustrate the product display problem and its possible solutions. In this case, the solution consists of a 2-period time horizon. In each period, there are 3 display tables with a capacity of 5 items each. Each display table and item is represented by its identifier (ID). Hence, for instance, in period h_1 , display table 2 will be composed of items 11, 29, 45, 18, and 5.

When selecting a set of products to be displayed, one should always consider customer preferences and willingness to buy (Choi et al., 2006). Current online display systems provide a list of products which are either based on the user's past behavior or on decisions made by similar users. This strategy has been widely applied in e-commerce, where it has generated raised sales as well as customer satisfaction (Kaminskas et al., 2017). Companies such as Amazon use a method called collaborative filtering. Here, ratings and purchases made by similar users are considered to suggest products to online customers (Ahn, 2008).

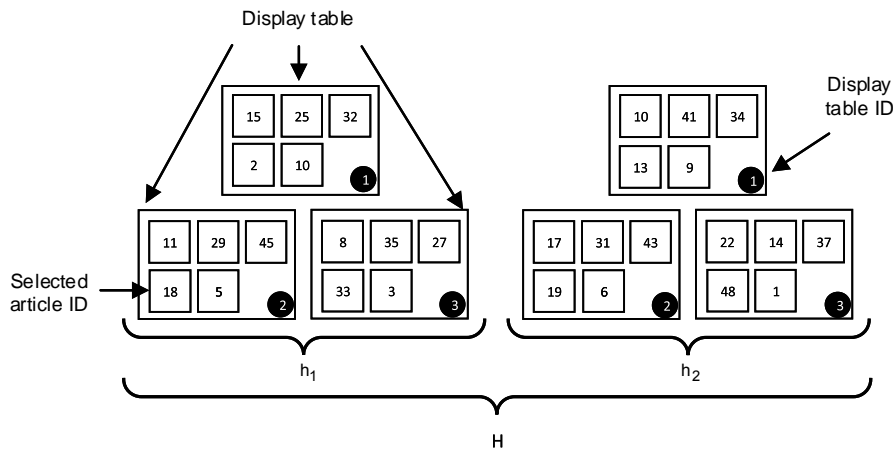


Figure 5.1: A solution representation for a simple multi-period product display problem.

This product selection problem is also relevant for brick-and-mortar stores, since they might benefit from an optimal selection of products to be displayed over a multi-period time horizon. The main contributions of this Section are described next: (i) a novel mathematical formulation for the multi-period product display problem with dynamic attractiveness levels is proposed in order to clearly define the problem under consideration –while analyzing a case study, the assumptions of this model were discussed with professionals of the retail sector, who were also students in an MBA offered at our business school; (ii) in order to solve this optimization problem in the context of a retail store with several display tables, biased-randomized (BR) versions of the greedy randomized adaptive search procedure (GRASP) and the iterated local search (ILS) are introduced; (iii) a set of novel benchmark instances, considering realistic constraints and different product characteristics, is proposed to test the quality of our approach when compared with non-linear solvers; and (iv) based on the outcomes of our experiments, a series of practical recommendations are provided. A complete introduction to biased-randomization techniques can be found in Grasas et al. (2017). A review on GRASP algorithms can be found in Festa and Resende (2009), while a description of the ILS metaheuristic framework can be found in Lourenço et al. (2003). Finally, a recent study on the combination of biased-randomization techniques with GRASP is available in Ferone et al. (2018). Regarding the constraints considered in this work, they include diversity of fashion collections, selling-price categories, and marginal-profit categories. Likewise, dependencies across both periods and items are considered in our study.

5.1.1 Literature Review

This section has been divided into two sections, each dealing with different problems strongly related to the product display problem considered in this section.

5.1.1.1 The Product Assortment and Product Recommendation Problems

The product assortment problem involves finding the optimal combination of products to include in a limited portfolio of items to be manufactured or sold. One of the first works on the product assortment problem is due to Sadowski (1959). Since then, many researchers have explored this field, as illustrated in the survey made by Pentico (2008). This author classified the literature according to the following characteristics: demand, demand pattern, dimensions, number of stock sizes, substitution cost structure, and stocking pattern stability. In this sense, different authors studied the way customers make decisions and how that affects the stock offered or shown. Mahajan and Ryzin (2001) considered that customers choose only among products that are still in stock. Caro et al. (2014) and Ulu et al. (2012) developed models in which the assortment needs to be adapted over time. As identified by Mantrala et al. (2009), product assortment not only has the constraints of physical space and retailers' budget, but an attractiveness factor –as perceived by customers– should also be taken into account. Caro et al. (2014) considered a problem in which the attractiveness of products decays over time once they are introduced to the selected assortment. A related study on space and store operations is provided by Mou et al. (2017), who considered how product attractiveness decreases with time. They also discuss the need for retailers to gather information from different channels in order to better plan their stock assortment.

Related to this, Honhon et al. (2010) studied product substitution after a stock-out of a first-choice item occurs. Here, customers choose the products that are available at the time of their visit to a physical store. Similarly, other authors analyzed the management of multi-item retail inventory systems with demand substitution (Smith and Agrawal, 2000) and the dynamic assortment planning with demand learning (Sauré and Assaf, 2013). According to the former authors, profitability depends on incorporating substitution effects in inventory management. Substitution increases the demand for other items and affects optimal stock levels. According to the latter authors, it is vital for retailers to select which products to offer due to the limited display capacity in the physical stores. Hence, these authors described different stock assortment policies and introduce a model for dynamic assortment planning. Similarly, Honhon et al. (2010) proposed a dynamic programming algorithm to determine the optimal assortment in a single-period problem with stock out-based substitution. Rajamma et al. (2007) described a method for determining inventory depth and variety breadth, as well as the mix between basic and seasonal clothing in fashion retail. Strategic decisions on the right variety and depth of in store stock have been developed by Mantrala et al. (2009). They provided reviews on how to customize retail assortment at the store level, rather than simply using a centrally planned assortment for all stores. A complete review on stock assortment is provided by Kök et al. (2008).

The product recommendation problem has received increasing attention recently. According to Liu and Shih (2005), "recommend systems rely on customer purchase history to determine customer preferences and to identify products that customers may purchase." Li et al. (2014b) proposed a framework for a localized product recommendation system associated with automatic vending machines. Their system offers suitable recommendations

of localized products to customers in different locations. They developed a hybrid technique using a metaheuristic approach, a clustering technique, as well as classification and statistical methods. The importance of product recommendation in today's omnichannel retailing world is also mentioned by Balakrishnan et al. (2018). They adopted an intuitive co-clustering algorithm for locating useful patterns in a *0-1 matrix*, which studies the buying behavior of customers using historical data on past purchases. In order to handle the product recommendation problem for e-commerce applications, Baykal et al. (2005) proposed a co-operation framework for multiple role-based reasoning agents. Choi and Cho (2004) presented a similar product-finding algorithm for the collaborative business companies that share a product taxonomy table and have exchangeable products information. Choi et al. (2012) proposed an online product recommendation system, which combines implicit rating-based collaborative filtering (CF) and sequential pattern analysis (SPA). The system derives implicit ratings by applying CF to online transaction data –even when no explicit rating information is available–, and integrates CF and SPA for improving recommendation quality.

Zhao et al. (2014) developed a novel product recommender system called METIS. Their system identifies, almost in real time, users' purchase trials from their microblogs. Then, it makes product recommendations based on matching the users' demographic information –extracted from their public profiles– with product demographics learned from these microblogs and additional online reviews. Zhao et al. (2016) proposed a novel solution for 'cross-site' and 'cold-start' product recommendation, which recommends products from e-commerce websites to users at social networking sites in cold-start situations. The term cold-start refers to users who do not have historical records on the items they have purchased. The authors proposed learning both users' and product feature representations via data collected from e-commerce websites, using recurrent neural networks and then applying a modified gradient boosting trees method to transform users' social networking features into user purchase preferences. More recently, Kaminskas et al. (2017) addressed a particular product recommendation problem regarding small-scale retail websites, where the small number of returning customers makes traditional user-centric personalization techniques inapplicable. Hence, these authors applied an item-centric product recommendation strategy that combines two well-known methods –association rules and text-based similarity– for generating recommendations based on a single 'seed' product. Furthermore, their approach was also used to recommend products based on a set of seed products in a user's shopping basket. The effectiveness of their recommendation approach is demonstrated, in the product-seeded and basket-seeded scenarios, through a series of experiments employing real customer data.

Product recommendation systems are related to the product assortment problem: a set of correlated products must be selected to be exposed (or recommended) in an exposition area with limited capacity. This selection of products should help to improve the experience of customers when visiting a store. In effect, by exposing an appropriate set of items at the display tables it is possible to increase the level of attraction of customers to the store, which directly influences the customers' experience and, hence, the sales revenue. Although this

Section shows that research has been undertaken on both product recommendation systems and the product assortment problem, there is still a lack of work on how to combine different products in retail display tables at brick-and-mortar stores, especially when considering a multi-period time horizon and dynamic attractiveness values.

5.1.1.2 The Shelf Space Allocation Problem

A related issue is the shelf space allocation problem (Yang and Chen, 1999; Yang, 2001), which is also linked to the optimal selection of products to be displayed in shelves with limited space. In this problem, however, several items of the same product can be selected. According to Hübner and Kuhn (2012), there are two different questions associated with the planning: assortment planning and shelf space planning. The first relates to listing decisions based on consumer choice, whereas the second deals with the limited shelf space. Our work considers both aspects, that is, what products should be shown in a limited space. Retail stores, especially brick-and-mortar ones, need to make decisions on what stock should be displayed in order to increase the customer's attention. Despite the proliferation of numerous software applications in shelf space management, which make use of historical data, new algorithms –as the ones introduced in this section– are needed to deal with multi-period and dynamic versions of the product selection problem. Most of the literature about the shelf space allocation problem refers to supermarket products (Flamand et al., 2016; Bianchi-Aguiar et al., 2016). Other areas, such as fashion stores, have not received much attention. Parsons (2011) analyzed how the atmosphere in fashion stores influences sales. Gao et al. (2014) investigated how pre-packs are used in retail distribution and how this reduces handling costs. Notice that fashion and apparel customers do not approach the store with a clear purchase objective, as it would be the case for food in supermarkets or electronic equipment. Instead, the customer enters the store just to look and see if something attracts his / her attention. It is the store's function to appeal to them to make them buy from the store and not from another shopping channel, e.g., a mobile device, tablet, or personal computer.

Table 9.1 summarizes the literature on product recommendation, product assortment, and shelf space allocation problems according to the solving methodology employed. Among them, we highlight the use of dynamic programming methods, heuristics / metaheuristics, and data mining / machine learning strategies such as collaborative filtering, association rules, and sequential pattern analysis. From the literature review, one can notice that approximate methodologies have gained more popularity over the years. This fact can be explained by the continuous growth in the size of realistic instances.

5.1.2 Problem Definition

Consider a warehouse holding a set of products or items. It has to supply a retail store for different time periods, which defines the planning horizon. Each item belongs to a certain collection (e.g., shirts or jeans, in the case of clothes), has a selling price –which might vary with time–, and a marginal profit –which is typically given as a percentage of the selling price. Depending on its selling price, an item is classified as 'expensive' or not. In any

Table 5.1: Classification of main articles by problem and solving methodology.

Study	Problem			Methodology		
	Product Recommendation	Assortment Problem and Stock Optimization	Shelf Space Allocation	Exact	Heuristic or Metaheuristic	Other
Sadowskit (1959)		•		•		
Mahajan and Ryzin (2001)		•				•
Choi and Cho (2004)	•				•	
Baykal et al. (2005)	•				•	
Liu and Shih (2005)	•				•	•
Honhon et al. (2010)		•		•		
Choi et al. (2012)	•					•
Sauré and Assaf (2013)		•		•		
Caro et al. (2014)		•		•	•	
Gao et al. (2014)		•				•
Li et al. (2014b)	•				•	•
Zhao et al. (2014)	•					•
Bianchi-Aguiar et al. (2016)			•		•	
Flamand et al. (2016)			•		•	
Kaminskas et al. (2017)	•					•
Balakrishnan et al. (2018)	•					•

given period, the retail store contains a set of tables, each of them displaying a subset of non-repeated items. Each item has an initial attractiveness value, estimated from experts' opinion and/or historical observations in an omnichannel environment –such as the number of times it has been selected and analyzed in the past, the feedback provided by the customers, etc. The attractiveness value can also depend upon other items currently being displayed in the table, since relations (or dependencies) between pairs of products may need to be considered.

Among all the available products in the warehouse, a subset of different items should be selected to be displayed on the retail display tables. The dependency between each pair of items is registered in a dependency matrix. An inter-period dependency is also considered. The attractiveness value of each item is reduced by a known quantity –typically expressed as a percentage– every time the product is repeatedly shown in two (or more) consecutive periods. In other words, if an item is repeatedly exposed during several consecutive periods, its novelty disappears and, as a consequence, its attractiveness value is reduced. On the other hand, whenever an item has not been shown in the previous period, its attractiveness value is increased due to the novelty effect. The goal is then to solve a multi-period product display problem with dynamic attractiveness levels. In this problem, a subset of items has to be selected to be displayed at each table-period combination in order to maximize the aggregated attractiveness level over all periods. In order to make the problem more realistic, some additional constraints are also considered in this section:

1. *Collection constraint:* the subset of items assigned to each table should cover at least a minimum number of goods from each collection, $l_c \geq 0$.
2. *Price constraint:* a minimum number of products at each table, $l_p \geq 0$, should belong to the expensive category.

3. *Profit constraint*: the profit margin of each table should be greater than a threshold defined by the manager, $l_m \geq 0$.

5.1.2.1 Mathematical Model

A set of consecutive time periods H is considered, together with a finite set of items, I , which are hosted in a warehouse. Each item $i \in I$ is associated with a base price $p_i > 0$, a marginal benefit $m_{ih} \geq 0$ (which might be different at each period $h \in H$), and an initial attractiveness value $v_{i0} > 0$. The final selling price of each item $i \in I$ at period $h \in H$ is given by $p'_{ih} \geq p_i$, i.e.: $m_{ih} = p'_{ih} - p_i$. These items belong to a set of collections $C = \{c_1, c_2, \dots, c_{|C|}\}$, where $I = \bigcup_{k=1}^{|C|} c_k$. The subset of expensive items is given by $I_p = \{i \in I / p_i \geq p_0\}$, where p_0 is a threshold price value defined by the manager.

At each period $h \in H$, a subset S_h items must be exposed using a set of homogeneous tables T , with each table containing a total of $n > 0$ items. The decision variable x_{ith} is equal to 1 if item i is selected for table t in period h , and to 0, otherwise. Thus, the set of non-repeated selected items for each table $t \in T$ in period $h \in H$ is given by $S_{th} = \{i \in I / x_{ith} = 1\}$, being $S_h = \bigcup_{t \in T} S_{th}$. A matrix $D = [d_{ij}]_{i,j \in I}$ provides the existing interaction value, $d_{ij} \in \mathbb{R}$, between any pair of items $i, j \in I$. These intra-period dependencies account for the fact that some items might be positively or negatively correlated with others (i.e., showing them together might generate synergies or, on the contrary, might reduce their aggregated attractiveness). Apart from this intra-period dependencies between pairs of items, inter-period dependencies are also considered to account for the product's novelty (or the lack of it). Accordingly, the attractiveness value of every item is a dynamic input, i.e., it is reduced or increased by a certain percentage factor depending on whether the item was displayed or not in the previous period. Thus, $\forall h \in \{1, 2, \dots, |H|\}$, the attractiveness value of item i in period h , v_{ih} , is recursively defined as:

$$v_{ih} = f(v_{i(h-1)}, x_{it(h-1)}, a, b, u, w) = \begin{cases} \text{Max}\{a, (1 - u)v_{i(h-1)}\} & \sum_{t \in T} x_{it(h-1)} = 1 \\ \text{Min}\{b, (1 + w)v_{i(h-1)}\} & \sum_{t \in T} x_{it(h-1)} = 0 \end{cases}$$

where $0 \leq a < b$ are bounds for the attractiveness values and $u, w \geq 0$ are decreasing or increasing percentage factors, respectively.

With this notation, the addressed problem can be formulated as follows:

$$\text{Max} \sum_{h \in H} \sum_{t \in T} \sum_{i \in I} v_{ih} x_{ith} + \sum_{h \in H} \sum_{t \in T} \sum_{i \in I} \sum_{j \in I / i < j} d_{ij} x_{ith} x_{jth} \quad (5.1)$$

Subject to:

$$\sum_{i \in I} x_{ith} = n \quad \forall t \in T, \forall h \in H \quad (5.2)$$

$$\sum_{t \in T} x_{ith} \leq 1 \quad \forall i \in I, \forall h \in H \quad (5.3)$$

$$\sum_{i \in c_k} x_{ith} \geq l_c \quad \forall c_k \in C, \forall t \in T, \forall h \in H \quad (5.4)$$

$$\sum_{i \in I_p} x_{ith} \geq l_p \quad \forall t \in T, \forall h \in H \quad (5.5)$$

$$\sum_{i \in I} m_{ih} x_{ith} \geq l_m \quad \forall t \in T, \forall h \in H \quad (5.6)$$

$$v_{ih} = f(v_{i(h-1)}, x_{it(h-1)}, a, b, u, w) \quad \forall i \in I, \forall h \in H \setminus \{0\} \quad (5.7)$$

$$x_{ith} \in \{0, 1\} \quad \forall i \in I, t \in T, h \in H \quad (5.8)$$

$$v_{ih} \in [a, b] \quad \forall i \in I, h \in H \quad (5.9)$$

The objective function (7.1) maximizes the total attractiveness of the planning horizon by considering the individual attractiveness of the items and the intra-period dependencies between each pair of selected items in each displaying table and period. Equation (5.2) ensures that the number of items on each table $t \in T$ does not exceed a pre-defined value n . Equation (7.2) guarantees that each item i cannot be selected more than once in a given period $h \in H$. Equation (7.3) confirms that, inside each period, each table covers at least l_c items from each collection c_k . Equation (7.4) guarantees that, inside each period, each table contains at least l_p expensive items. Equation (7.5) ensures, for each period, that the profit margin of each table should be greater than l_m . Equation (5.7) introduces the inter-periods dynamic component in the attractiveness value of the items. Notice that this equation transforms the objective function into a non-smooth one due to the definition of the v_{ih} values. Equation (5.8) states that all decision variables are binary. Finally, Equation (5.9) bounds the values that variable v_{ih} can take. A ‘relaxed’ version of this problem can be obtained when no bounds are imposed on the attractiveness values of each item. In that case, the objective function becomes quadratic since Equation (5.10) can be rewritten as:

$$v_{ih} = v_{i(h-1)}(1 - u) \sum_{t \in T} x_{it(h-1)} + v_{i-1}(1 + w) \sum_{t \in T} (1 - x_{it(h-1)}) \quad \forall i \in I, \forall h \in H \setminus \{0\} \quad (5.10)$$

5.1.3 Solution Method: a GRASP and an ILS Approach

In this Section, biased-randomized versions of the well-known GRASP (Feo and Resende, 1995) and the ILS (Lourenço et al., 2003) metaheuristics are proposed to solve the multi-period product display problem with dynamic attractiveness. Both algorithms consist of some common stages: (i) a construction stage, in which a feasible initial solution is built taking into account the constraints; and (ii) an improvement stage, in which a local search

is applied to the initial solution in order to enhance its quality. Apart from these two common stages, the ILS incorporates a perturbation phase and an acceptance criterion phase. As discussed in Resende and Ribeiro (2016) and Graspas et al. (2016), both GRASP and ILS are relatively easy-to-implement and flexible metaheuristic frameworks that have shown their efficiency in solving different optimization problems, including both deterministic and stochastic ones. Also, they typically do not require many parameters or time-consuming fine-tuning processes. The previous properties make them especially suitable for industrial applications. Moreover, they have been successfully combined with biased-randomization techniques in multiple occasions (De Armas et al., 2017; Ferone et al., 2018; Ferrer et al., 2016; Gonzalez-Neira et al., 2017).

In the BR-GRASP approach, a solution is built iteratively, element by element, and then improved via a local search procedure. This two-step process is repeated until a number of iterations (or a maximum running time) is reached. Then, the best-found solution is returned. On the other hand, the BR-ILS starts from a base solution, which is repeatedly perturbed (modified using a destruction-reconstruction process), enhanced via a local search procedure, and finally evaluated by an acceptance criterion until a stop condition is met. In both the BR-GRASP and the BR-ILS, a partial solution is constructed for each new period (taking into account the current configuration of the display tables). Then, this partial solution is improved through a local search procedure. The low-level details of these algorithms are provided next.

5.1.3.1 Extending the GRASP and ILS to a Biased-Randomized Algorithms

As mentioned, the use of Monte Carlo sampling techniques to enhance the performance of constructive heuristics was proposed by Faulin and Juan (2008). In the proposed approach, more advanced biased-randomized techniques –based on the use of a geometric probability distribution– are used every time a new solution is constructed or partially reconstructed after a perturbation phase. BR techniques differ from standard selection strategies, which are usually based on a greedy criterion or on the use of a uniform probability distribution to select the next candidate from a list. Thus, for example, in a classical GRASP framework, a RCL is considered, and a uniform probability distribution is used to choose a candidate from this RCL. However, in a BR-GRASP, an unrestricted candidates list (UCL) is employed. This UCL is sorted according to some logical criterion, and a geometric distribution is used to select the next element from this sorted UCL (Ferone et al., 2018). The geometric distribution uses a single parameter, β , which is proportional to the probability of selecting the first element in the UCL. Also, all elements in the UCL receive a probability of being selected, which is higher the closer the element is to the top of the sorted UCL. The same concept is also employed during the solution-construction processes inside our BR-ILS algorithm.

Pseudocode 13 illustrates this solution-construction process. All items (set I) are included in the UCL, which is sorted in descending order according to the ‘adjusted’ attractiveness value of each item, i.e.: the original attractiveness value in the corresponding period is corrected to consider dependencies with respect to other items already in the display table. Then, the geometric distribution is used to randomly build a solution by selecting one

‘promising’ item at a time. Once selected, the element is removed from the UCL, the adjusted attractiveness values and profit margin of the remaining elements are updated, and the UCL is sorted again. The efficiency of similar BR strategies has been extensively discussed in different studies, such as Dominguez et al. (2014) and Dominguez et al. (2016a), Juan et al. (2015b), and Martin et al. (2016)

Pseudocode 13: Biased-Randomized Construction Stage

Data: set of items I , geometric distribution parameter β

```

1 Function constructBiasedRandomizedSolution( $I, \beta$ ):
2    $s \leftarrow \emptyset$ 
3   Initialize candidate list set:  $UCL \leftarrow I$ 
4   Order UCL according to sorting criterion  $c(\cdot)$ 
5   while solution  $s$  is not complete do
6     Randomly select  $pos \in \{1, \dots, |UCL|\}$  according to distribution  $Geom(\beta)$ 
7      $s \leftarrow s \cup \{UCL[pos]\}$ 
8      $UCL \leftarrow UCL \setminus \{UCL[pos]\}$ 
9     Re-order UCL
10  end
11  return  $s$ 
12 End

```

5.1.3.2 Constructing an Initial Solution for the Current Period

For each period in the planning horizon, an initial solution is built by adding products to the display tables. We assume that these tables are empty at the beginning of the period. Also, each product is assumed to have an initial attractiveness value at the beginning of the first period. This attractiveness value is based on historical observations and, possibly, expert judgment.

Using the BR strategy previously described, a subset of items is assigned to a given display table. First, a subset S_{th} is built by selecting n products for table t in period h . At this point, the collection constraint is incorporated into the construction procedure by selecting a minimum number of items from each collection. Next, the price-related constraint and the profit-margin constraint are checked for the S_{th} subset. If both constraints are satisfied, the next table is considered. Otherwise, the solution is repaired. During the repair process, an item is randomly selected from S_{th} and replaced by another item not in S_{th} . In the case of the price constraint, for instance, this swapping process is based on the replacement of items from one price category (e.g., non-expensive) by products belonging to another one (e.g., expensive). In the case of the profit-margin constraint, the swapping process is based on the replacement of items with a low profit margin by products with a high profit margin. This process is repeated until a feasible configuration of tables is eventually achieved for the current period. After obtaining a feasible configuration for the current period, an improvement stage is applied as described next.

5.1.3.3 Improving the Solution of the Current Period

Pseudocode 14 illustrates the improvement procedure applied to each table in the current period. It starts from the first table t in the given period h . An item i is randomly selected from t and removed. Then it is replaced by another randomly-selected item $j \in I$ among those that can be inserted without violating any constraint. As a result, a new table t' is generated. The adjusted attractiveness value of t' is updated taking into account the dependencies between pairs of items. If the adjusted attractiveness value of t' is greater than that of t , the latter table is updated and the counter is reset. Otherwise, another item is randomly chosen until a maximum number of iterations is achieved without any improvement. The same process is applied to the remaining tables in the current period h . At each iteration of the constructive procedure, the articles' attractiveness and profit margins are conveniently updated for the next periods of the planning horizon.

Pseudocode 14: Refinement of the Configuration of Tables

Data: period h , maximum number of iterations without improvement $iter_{max}$

```

1 Function localSearch( $h, iter_{max}$ ):
2    $articleToRemove \leftarrow null$ 
3    $articleToInsert \leftarrow null$ 
4   for each table  $t \in T$  of  $h$  do
5      $counter \leftarrow 0$ 
6     while  $counter < iter_{max}$  do
7        $t' \leftarrow t$ 
8        $itemToRemove \leftarrow$  random item  $i \in S_{t'h}$ 
9        $itemToInsert \leftarrow$  random non-selected article  $j \in I$  that keeps the feasibility of  $t'$ 
10      Remove  $itemToRemove$  from  $t'$ 
11      Insert  $itemToInsert$  into  $t'$ 
12      Update total attractiveness and correlation of  $t'$ 
13      if  $tableAttractiveness(t') > tableAttractiveness(t)$  then
14         $t \leftarrow t'$ 
15         $counter \leftarrow 0$ 
16      else
17         $counter \leftarrow counter + 1$ 
18      end
19    end
20  end
21 End

```

5.1.3.4 Perturbation Stage in the BR-ILS

Both the construction of an initial solution and the improvement process are employed in the BR-GRASP and the BR-ILS. However, the BR-ILS also makes use of a base solution which is modified via a perturbation stage. In this case, the perturbation is characterized by a destruction-reconstruction process as described in Pseudocode 15. After selecting a starting period, h_{start} , all posterior periods are destroyed and then re-built, using the previously described constructive and improvement stages.

The starting period is chosen according to a varying percentage of destruction, $k \in (0, 1]$. Hence, for example, if $k = 0.1$ then the last 10% of the periods are destroyed and reconstructed (due to the dependencies across periods, if a period is rebuilt all posterior periods need to be recomputed).

Pseudocode 15: Perturbation Stage

Data: base solution $baseSol$, percentage of destruction k , number of periods $|H|$

```

1 Function perturbation( $baseSol, k, |H|$ ):
2    $h_{start} \leftarrow |H| - k * |H|$ 
3    $newSol \leftarrow baseSol$ 
4   for each period  $h \in H$  of  $newSol$  starting from  $h_{start}$  do
5      $newSol \leftarrow newSol \setminus \{h\}$ 
6      $h^* \leftarrow reconstruction(h)$ 
7      $h^* \leftarrow improvement(h^*)$ 
8      $newSol \leftarrow newSol \cup \{h^*\}$ 
9   end
10  if  $newSol$  is better than  $baseSol$  then
11     $baseSol \leftarrow newSol$ 
12  end
13 End

```

5.1.3.5 Acceptance Criterion Stage in the BR-ILS

Finally, the ILS metaheuristic also incorporates an acceptance criterion to reduce the probability of getting trapped in a local minimum (Pseudocode 16). In our case, we use the demon-based acceptance criterion described in Juan et al. (2014). In this criterion, newly generated solutions are compared with the base solution, and the former is updated in two cases: (i) when the new solution is better than the base solution; or (ii) when the new solution is worse than the base solution but the difference in value is lower than the improvement (credit) obtained in the last update of the base solution.

Pseudocode 16: Credit-Based Acceptance Criterion

Data: base solution $baseSol$, new solution $newSol$

```

1 Function acceptance( $baseSol, newSol$ ):
2    $delta \leftarrow cost(newSol) - cost(baseSol)$ 
3   if  $delta \leq 0$  then
4      $credit \leftarrow -delta$ 
5      $baseSol \leftarrow newSol$ 
6     if  $cost(baseSol) < cost(bestSol)$  then
7        $bestSol \leftarrow baseSol$ 
8     end
9   end
10  if  $0 < delta \leq credit$  then
11     $credit \leftarrow 0$ 
12     $baseSol \leftarrow newSol$ 
13  end
14 End

```

5.1.4 Computational Experiments and Results

This section describes the experimental setup designed to evaluate the performance of our BR-GRASP and BR-ILS algorithms. To the best of our knowledge, this is the first work solving a rich and realistic version of the multi-period product display problem with dynamic attractiveness. Hence, we had to generate a complete set of benchmarks with different characteristics to comprehensively evaluate and test the proposed algorithms. These characteristics are: number of articles ($|I|$), number of display tables ($|T|$), number of collections ($|C|$), and number of items per display table (n).

For small-sized instances (aimed at being solved using non-linear exact methods), we set $|I| \in \{25, 50, 75, 100\}$, $|T| \in \{2, 3, 4\}$, $|C| = 5$, $|H| \in \{2, 3, 4\}$, and $n = 6$. Each of these instances was named according to these specifications. Thus, for example, instance *75i-5c-4p-4t-6it* consists of 75 items, 5 collections, 4 periods, 4 tables per period, and 6 items per table. Similarly, for the large-sized instances we set $|I| \in \{500, 1000, 1500, 2000\}$, $|T| \in \{5, 10\}$, $|C| = 4$, $|H| = 12$, and $n = 10$. Each of these instances was named according to these specifications. Thus, for example, instance *a500m5i1* consists of 500 products and 5 tables. The last index represents different instances using the same combination of items and tables. For the purpose of numerical experimentation, most of the specific inputs in these instances (e.g., initial attractiveness values, dependencies between pairs of products, item prices, and associated collection) have been randomly generated. In order to facilitate reproducibility of the experiments, all these instances and inputs are publicly available [online](#).

Although all input data is available in the previous link, an overview of these inputs is provided next for the case of the large-sized instances. The final selling price of each product is generated according to a uniform distribution in the range $[10, 150]$ (monetary units). The profit-margin percentage for each product follows a uniform distribution in the range $[10\%, 35\%]$. The price and profit margin are considered to generate the absolute profit, which is used to check whether the respective constraints are satisfied. The initial attractiveness value for each item and the between-items dependencies are generated according to a uniform distribution in the range $[10, 100]$ and $[-35, 35]$, respectively. Regarding the considered constraints, the subset of selected products at each table should cover at least 20% of each collection. Regarding prices, the items are categorized into two different categories: those with a cost inferior to 60 monetary units are considered as non-expensive items. The rest are considered as expensive. In our experiments, we required that the selected subset at each table should include at least 50% of expensive products. Finally, for each table, the profit margin per table is set at 100 monetary units or more. To account for the ‘novelty factor’, when a product is displayed on a table during a given period, its attractiveness value is decreased by 10% for the next period (always considering that the minimum attractiveness value that a product can reach is bounded by the modeling parameter $a \geq 0$). Conversely, if a product is not displayed at a given period, its attractiveness value increases by 3% for the next period (up to a maximum value given by the modeling parameter $b > a$).

After some initial tests, a geometric probability distribution with a parameter β randomly chosen in the interval $(0.80, 0.99)$ was used for the biased-randomization process

during the solution-construction stage. The stopping criterion for the BR-GRASP and BR-ILS is defined by a maximum computing time, t_{max} , defined as: $t_{max} = 0.5 \cdot |I| \cdot |T| \cdot |H|$. In practice, this represents approximately 15 seconds of execution for instances composed of 500 articles, 5 tables, and a planning horizon of 12 days, for example. Regarding the improvement stage, the stopping criterion is set to $it_{max} = 1000$ iterations without observing any improvement. Our algorithms were coded in Java and run on a standard PC with an Intel Core i5 CPU at 2.7 GHz and 8 GB RAM.

Tables 5.2 and 5.3 summarize the experimental setup and parameters for the computational experiments. Note that a new parameter, r , is introduced. This parameter refers to the percentage of profit margin (and price) reduction in specific sales periods.

Table 5.2: Problem Parameters

$ H $	h_s	h_e	r	u	w	l_c	l_p	l_m	b	a
12	11	12	10%	10%	3%	20%	50%	100	150	0

Table 5.3: Methodologies Parameters

α	β	t_{max}	it_{max}	k
0.1	$[0.8, 1)$	$0.5 \cdot I \cdot T \cdot H $	1,000	$\{0.1, 0.3, 0.4, 0.55, 0.7, 0.8, 1\}$

5.1.4.1 Analysis of Results

As discussed in Section 5.1.2, the relaxed version of the problem is non-linear. It might be solved using non-linear exact methods, at least up to a certain size. Hence, in order to compare the solutions generated by our BR-GRASP algorithm with the ones provided by the non-linear solver, we run a first set of small-sized instances. In a second experiment, the proposed large-sized instances are employed to test our methodologies under more realistic (large-scale and non-smooth) scenarios.

5.1.4.2 Small-sized Instances and Limitations of Non-linear Solvers

A set of small-sized instances was generated in order to test the performance of our BR-GRASP algorithm when compared to state-of-the-art non-linear solvers. As explained in the previous section, these instances differ in the number of articles, number of tables, and the length of the horizon. Table 5.4 presents the results of a numerical analysis discussing the limitations of these solvers even when dealing with the relaxed version of the problem. The overall BKS for each instance is provided. According to our experiments, only instances with up to 600 binary variables and 200 constraints (e.g., 50 items, 4 periods, and 3 tables per period) can be solved in reasonable computing times using modern non-linear solver engines such as COIN-OR Bonmin and NEOS Bonmin (Bonami and Lee, 2007). Notice that our BR-GRASP algorithm (BR-GR) is very competitive when compared with the non-linear solver engines, obtaining better or similar solutions in much shorter computing times.

Table 5.4: Comparison of results between the non-linear solvers and our BR-GRASP approach (relaxed version of the problem)

Instance	Instance details					GAP (%) w.r.t. BKS				Time (sec.)		
	I	H	T	bin var	constraints	BKS	COIN-OR	NEOS	BR-GR	COIN-OR	NEOS	BR-GR
25i-5c-2p-2t-6it	25	2	2	100	> 50	311.4	4.82	3.21	0.00	19	23	1
25i-5c-3p-2t-6it	25	3	2	150	> 75	440.4	2.04	2.68	0.00	10	23	1
25i-5c-4p-2t-6it	25	4	2	200	> 100	562.9	0.00	1.03	1.49	37	69	4
50i-5c-4p-2t-6it	50	4	2	400	> 200	768.7	0.91	0.00	0.85	196	46	5
50i-5c-4p-4t-6it	50	4	4	800	> 200	1250.3	0.00	2.62	2.58	> 700	> 300	3
75i-5c-2p-2t-6it	75	2	2	300	> 150	392.4	3.52	0.00	2.47	50	79	5
75i-5c-2p-3t-6it	75	2	3	450	> 150	557.7	3.39	1.27	0.00	148	165	5
75i-5c-3p-3t-6it	75	3	3	675	> 225	1033.9	N/A	N/A	0.00	> 3600	> 1200	64
75i-5c-3p-4t-6it	75	3	4	900	> 225	1352.1	N/A	N/A	0.00	> 3600	> 1200	83
100i-5c-2p-3t-6it	100	2	3	600	> 200	583.1	N/A	1.73	0.00	> 3600	> 120	34
100i-5c-3p-2t-6it	100	3	2	600	> 300	631.6	0.24	0.00	1.35	> 700	> 300	4
100i-5c-3p-4t-6it	100	3	4	1200	> 300	1122.7	N/A	N/A	0.00	> 3600	> 1200	64
100i-5c-4p-4t-6it	100	4	4	1600	> 400	1496.9	N/A	N/A	0.00	> 3600	> 1200	61

5.1.4.3 Large-sized Instances with Fixed Profit Margins

Due to the satisfactory performance of our biased-randomized approach in small-sized instances, the BR-GRASP and the BR-ILS were also tested in solving more realistic (large-scale) instances of the complete (non-smooth) version of the problem. In this case, no variations in the selling price of the products were considered. A lower bound on the attractiveness value any item can achieve is imposed. In our experiments, we set $a = 0$, while b takes a sufficiently large value (i.e., in practice the attractiveness value of an item grows after each period in which the item has not been displayed). For each of the 40 generated instances, 10 runs were performed (each run using a different seed for the pseudo-random number generator). To evaluate the performance of each methodology, we consider the percentage *gap* between the best-found solution using that methodology (i.e., the one with the highest attractiveness value) and the BKS obtained with any solution methodology. Thus, the lower the gap, the better the performance of the methodology.

The solutions generated by our BR-GRASP and BR-ILS algorithms are compared with those generated by a greedy strategy, a ‘standard’ GRASP, and a ‘standard’ ILS (i.e., without considering BR techniques). The greedy methodology approximates human behavior (e.g., an expert manager) when selecting articles to be displayed. In the standard GRASP algorithm, and after some preliminary tests, the parameter α which controls the size of the RCL was set to 0.1. The standard ILS algorithm uses a greedy selection technique to build solutions. Regarding the perturbation stage, the destruction rate was defined by the set $k \in \{0.1, 0.3, 0.4, 0.55, 0.7, 0.8, 1\}$.

Table 5.5 provides the summarized results of the solutions generated by the greedy (G), BR-GRASP (BR-GR), GRASP (GR), ILS and BR-ILS methodologies. For each instance and methodology, the following data is provided: the attractiveness value of the best-found solution, the average (AVG) attractiveness value obtained after 10 runs, as well as its corresponding standard error (SE), the gap with respect to the BKS, and the average CPU time employed to find the corresponding solutions. The best-found solution for each instance is

presented in bold.

Table 5.5: Comparison of the results obtained by the proposed methodologies.

Instance	Best Solution (attractiveness)						AVG Attractiveness (SE)						GAP (%) wr.t. BKS						AVG Time (sec.)						
	G	BR-GR	GR	ILS	BR-ILS	G	G	BR-GR	GR	ILS	BR-ILS	G	G	BR-GR	GR	ILS	BR-ILS	G	G	BR-GR	GR	ILS	BR-ILS	G	
a500m5i1	100,303	104,887	104,726	103,882	104,447	100,303	(0)	104,540	(71)	104,418	(75)	103,676	(46)	104,131	(63)	4.37	0.00	0.15	0.96	0.42	0	8	7	10	12
a500m5i2	101,106	105,600	105,700	104,344	105,077	101,106	(0)	105,295	(59)	105,258	(83)	104,143	(53)	104,903	(48)	4.35	0.09	0.00	1.28	0.59	0	7	7	11	9
a500m5i3	100,313	104,955	104,721	103,930	104,688	100,313	(0)	104,596	(76)	104,506	(50)	103,614	(51)	104,295	(59)	4.42	0.00	0.22	0.98	0.25	0	8	7	12	12
a500m5i4	100,606	105,026	104,834	103,887	104,597	100,606	(0)	104,735	(59)	104,589	(48)	103,531	(61)	104,290	(59)	4.21	0.00	0.18	1.08	0.41	0	8	7	11	14
a500m5i5	100,633	105,745	105,700	104,647	105,018	100,633	(0)	105,293	(88)	105,165	(84)	104,127	(102)	104,701	(54)	4.83	0.00	0.04	1.04	0.69	0	7	8	10	10
a500m10i1	188,711	196,334	195,685	194,764	195,811	188,711	(0)	195,901	(106)	195,382	(77)	194,194	(113)	195,440	(98)	3.88	0.00	0.33	0.80	0.27	0	15	14	20	23
a500m10i2	189,668	197,098	196,662	195,567	196,288	189,668	(0)	196,734	(99)	196,347	(69)	195,086	(95)	195,957	(57)	3.77	0.00	0.22	0.78	0.41	0	14	15	21	20
a500m10i3	187,187	194,977	194,415	193,682	194,283	187,187	(0)	194,538	(108)	193,989	(66)	193,063	(111)	193,863	(65)	4.00	0.00	0.29	0.66	0.36	0	15	16	18	22
a500m10i4	187,302	196,927	195,653	195,125	195,885	187,302	(0)	196,421	(104)	195,232	(107)	194,526	(106)	195,611	(52)	4.89	0.00	0.65	0.91	0.53	0	15	15	18	20
a500m10i5	186,074	193,498	192,934	192,147	192,835	186,074	(0)	193,147	(83)	192,646	(77)	191,537	(116)	192,415	(93)	3.84	0.00	0.29	0.70	0.34	0	15	15	17	21
a1000m5i1	110,916	113,781	112,947	113,166	113,851	110,916	(0)	113,602	(45)	112,708	(58)	112,958	(58)	113,369	(73)	2.58	0.06	0.79	0.60	0.00	0	15	14	15	24
a1000m5i2	111,172	114,672	113,486	113,737	114,233	111,149	(0)	114,306	(53)	113,263	(37)	113,221	(107)	114,034	(32)	3.05	0.00	1.03	0.82	0.38	0	16	14	20	18
a1000m5i3	110,216	114,088	112,895	112,966	113,453	110,216	(0)	113,675	(73)	112,515	(66)	112,717	(50)	113,266	(58)	3.39	0.00	1.05	0.98	0.56	0	16	16	23	21
a1000m5i4	110,958	114,182	113,100	113,060	113,871	110,958	(0)	113,907	(71)	112,770	(61)	112,841	(51)	113,437	(65)	2.82	0.00	0.95	0.98	0.27	0	16	16	25	24
a1000m5i5	110,969	114,362	113,164	113,054	113,894	110,969	(0)	113,998	(68)	112,859	(52)	112,851	(44)	113,561	(51)	2.97	0.00	1.05	1.14	0.41	0	15	15	15	24
a1000m10i1	212,437	218,018	215,554	216,393	217,147	212,437	(0)	217,608	(86)	214,770	(117)	215,818	(110)	216,752	(80)	2.56	0.00	1.13	0.75	0.40	0	32	30	36	48
a1000m10i2	209,689	215,681	212,682	214,212	215,081	209,689	(0)	215,443	(87)	212,175	(83)	213,865	(67)	214,456	(114)	2.78	0.00	1.39	0.68	0.28	0	30	33	54	36
a1000m10i3	211,045	216,982	214,111	215,294	216,723	211,045	(0)	216,443	(89)	213,544	(94)	214,809	(107)	215,976	(111)	2.74	0.00	1.32	0.78	0.12	0	34	30	38	47
a1000m10i4	210,869	216,560	213,698	215,304	216,233	210,869	(0)	216,190	(109)	213,133	(123)	214,477	(112)	215,721	(73)	2.63	0.00	1.32	0.58	0.15	0	28	30	31	39
a1000m10i5	212,916	217,943	214,600	216,719	217,337	212,916	(0)	217,506	(84)	214,376	(58)	216,105	(144)	216,980	(64)	2.31	0.00	1.53	0.56	0.28	0	29	31	51	41
a1500m5i1	114,511	117,763	115,996	116,406	117,318	114,511	(0)	117,377	(74)	115,742	(50)	116,203	(69)	117,021	(67)	2.76	0.00	1.50	1.15	0.38	0	23	22	30	40
a1500m5i2	116,116	119,709	117,457	118,930	119,328	116,116	(0)	119,422	(42)	117,272	(32)	118,341	(114)	119,007	(63)	3.00	0.00	1.88	0.65	0.32	0	22	21	34	29
a1500m5i3	115,237	118,749	116,448	117,247	118,234	115,237	(0)	118,250	(72)	116,501	(32)	117,069	(43)	117,972	(56)	2.96	0.00	1.77	1.26	0.43	0	23	23	35	38
a1500m5i4	115,523	118,600	116,448	117,569	118,433	115,523	(0)	118,250	(72)	116,241	(56)	117,150	(84)	117,886	(69)	2.59	0.00	1.81	0.87	0.14	0	22	22	33	32
a1500m5i5	115,004	118,189	116,218	116,955	117,785	115,004	(0)	117,846	(76)	115,926	(57)	116,755	(69)	117,530	(54)	2.69	0.00	1.67	1.04	0.34	0	23	20	34	33
a1500m10i1	223,333	228,339	223,706	227,220	228,394	223,333	(0)	227,788	(119)	223,078	(93)	226,430	(125)	227,530	(101)	2.22	0.02	2.05	0.51	0.00	0	46	47	63	64
a1500m10i2	222,677	227,425	222,844	226,245	226,910	222,677	(0)	226,926	(96)	222,310	(78)	225,629	(113)	226,470	(98)	2.09	0.00	2.01	0.52	0.23	0	46	42	51	60
a1500m10i3	222,409	227,296	222,936	225,490	226,682	222,409	(0)	226,912	(119)	222,376	(109)	224,955	(96)	226,308	(91)	2.15	0.00	1.92	0.79	0.27	0	42	45	62	71
a1500m10i4	223,322	228,323	223,507	226,937	228,157	223,322	(0)	228,088	(55)	223,156	(79)	226,553	(96)	227,556	(92)	2.19	0.00	2.11	0.61	0.07	0	45	47	62	60
a1500m10i5	220,092	226,651	222,442	225,253	225,993	220,092	(0)	226,298	(61)	222,058	(68)	224,645	(133)	225,547	(80)	2.89	0.00	1.86	0.62	0.29	0	42	45	64	62
a2000m5i1	117,467	121,200	118,831	119,871	120,632	117,467	(0)	120,624	(74)	118,228	(74)	119,624	(54)	120,326	(60)	3.08	0.00	1.95	1.10	0.47	0	30	31	50	52
a2000m5i2	118,295	120,996	118,308	120,190	120,820	118,295	(0)	120,696	(68)	118,014	(52)	119,833	(81)	120,330	(72)	2.23	0.00	2.22	0.67	0.15	0	32	28	49	37
a2000m5i3	118,151	121,349	118,769	120,134	121,062	118,151	(0)	121,057	(53)	118,535	(57)	119,733	(70)	120,696	(70)	2.64	0.00	2.13	1.00	0.24	0	29	34	39	44
a2000m5i4	118,308	121,848	119,468	120,835	121,471	118,308	(0)	121,489	(76)	119,064	(68)	120,500	(58)	121,168	(57)	2.90	0.00	1.95	0.83	0.31	0	31	31	40	44
a2000m5i5	118,841	121,966	119,270	120,975	121,487	118,841	(0)	121,675	(42)	119,002	(46)	120,589	(74)	121,295	(65)	2.56	0.00	2.21	0.81	0.39	0	30	29	37	46
a2000m10i1	231,988	235,713	229,199	234,075	235,099	231,988	(0)	235,256	(84)	228,886	(62)	233,659	(106)	234,768	(72)	1.58	0.00	2.76	0.69	0.26	0	60	59	85	79
a2000m10i2	230,797	235,691	228,935	233,606	234,478	230,797	(0)	234,750	(135)	228,532	(76)	233,133	(83)	234,044	(62)	2.08	0.00	2.87	0.88	0.51	0	59	62	75	71
a2000m10i3	230,359	235,430	229,583	234,145	234,834	230,359	(0)	235,323	(33)	229,172	(80)	233,493	(184)	234,577	(48)	2.15	0.00	2.48	0.55	0.25	0	62	65	85	91
a2000m10i4	229,622	234,369	228,526	232,734	234,181	229,622	(0)	234,055	(64)	228,248	(55)	232,485	(61)	233,604	(87)	2.03	0.00	2.49	0.70	0.08	1	58	59	69	80
a2000m10i5	229,995	234,959	228,768	232,876	234,346	229,995	(0)	234,618	(98)	228,429	(76)	232,538	(85)	233,976	(69)	2.11	0.00	2.64	0.89	0.26	0	63	63	102	85
AVG	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	2.98	0.00	1.41	0.83	0.31	0	28	28	39	40

From Table 5.5, we can notice that the BR-GRASP and the BR-ILS (in that order) outperform the other methodologies. In particular, the BR-GRASP provides the BKS in 37 out of 40 tested instances. Although the BR-GRASP was not able to find the BKS in some cases, it was able to find solutions with a minor deviation (less than 0.10%) from it. According to our experiments, the greedy methodology runs extremely fast, since this methodology does not incorporate a local search mechanism and it is not embedded inside a multi-start process. Comparing the remaining methodologies, all of them require similar computing times, although the BR-GRASP reaches high-quality solutions faster than other approaches.

Figure 5.2 shows, for each methodology, a boxplot of the percentage gaps with respect to the BKS. From this Figure, one can conclude that the greedy methodology –which could be assimilated to a human behavior–, represents by far the worst approach. In effect, it shows an average gap of about 3% with respect to the BKS, a gap that can grow up to 5% in some instances. Also, it can be seen that biased-randomized techniques enhance the ILS and GRASP approaches. The main reason why our BR-GRASP performs slightly better than our BR-ILS might be the existence of inter-period dependencies. These inter-period dependencies might sometimes penalize partial-scope destruction-reconstruction processes (ILS) versus full-scope destruction-reconstruction ones (GRASP), since the former might get trapped more easily in a local minimum. Also, ANOVA and Fisher tests were run in order to analyze if the performance differences were statistically significant. As expected, the ANOVA test resulted in the existence of significant differences among the algorithms ($p\text{-value} = 0.000$). Actually, according to the Fisher pairwise test, except for the comparison between algorithms BR-GRASP and BR-ILS, all other differences in performance are statistically significant.

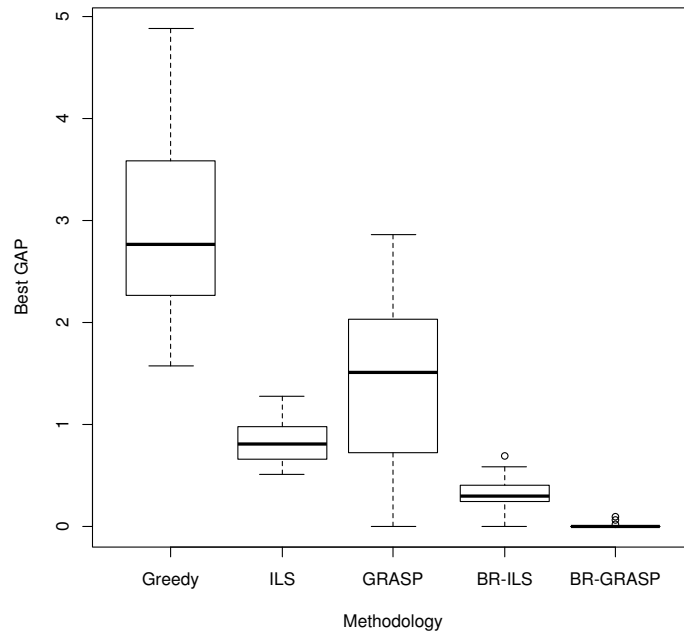


Figure 5.2: Comparison of percentage gaps w.r.t. the BKS achieved by each methodology.

Figure 5.3 illustrates the convergence behavior with time of the BR-GRASP algorithm for

an instance composed of 500 articles, 5 tables, and 5 periods. The convergence is analyzed by period P_i , $i \in \{1, 2, \dots, 5\}$, which runs for 1.25 seconds ($0.5 \times 500 \times 5$ milliseconds per period) for this problem. Notice that all periods present similar convergence behavior with time. Additionally, the range value of attractiveness is similar across the periods. This is explained by the small increment in attractiveness (3%) associated with new items, as well as by the large reduction in attractiveness (10%) associated with items that have lost their novelty effect.

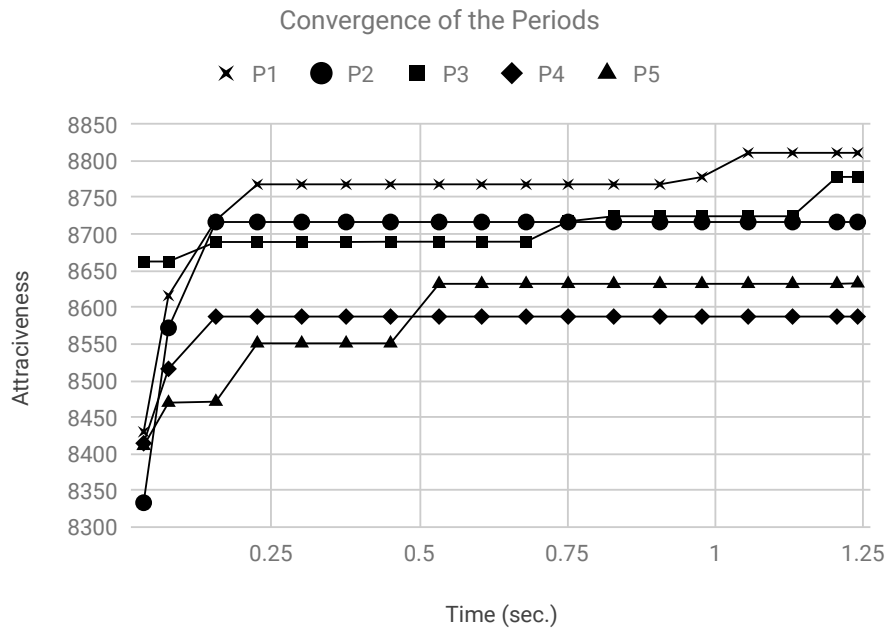


Figure 5.3: Convergence chart for the BR-GRASP in a particular instance.

5.1.4.4 Large-sized Instances with Varying Profit Margins (Selling Prices)

During the considered horizon, one could consider a reduction in product prices (and margins) due to the sales period. For running an experiment under these conditions, the parameters h_s and h_e , which represent the beginning and end of the especial sales period, were set to $h_s = 11$ and $h_e = 12$. The remaining parameters were not changed. Table 5.6 provides the summarized results of the solutions generated by our methodologies. For each problem magnitude, the first instance was considered, and a comparison among the solving methods was provided in terms of percentage gap.

From Table 5.6, one can notice that the BR versions of GRASP and ILS outperform the other solving methodologies. In this scenario with varying profit margin, the BR-GRASP is able to provide the BKS for the eight tested instances. For the remaining methodologies, their performance –in terms of gaps, SE, and CPU times– is similar to the one presented in Table 5.5.

5.1.5 Conclusions

Increasing levels of competitiveness among brands as well as among channels of the same brand make it difficult for retailers in brick-and-mortar stores to engage customers while in the shop. One of the ways to attract them to the stores is to offer a different experience and a factor of surprise. Displaying a set of correlated and attractive products on retail display tables that vary often is a promising way to engage customers with a pleasant experience. From a managerial perspective, being able to know the selection of products that maximizes the attractiveness level enables a rationalization of the stock available in the store. Moreover, reducing the time required to make these decisions might significantly increase productivity of the managers in charge of them.

In this section, we propose a rich and realistic multi-period product display problem, as well as biased-randomized algorithms that allow to solve it in an efficient way. In the considered problem, a set of correlated products has to be selected over multiple periods of time in order to maximize the total attractiveness level of the display tables in a retail store. A number of realistic characteristics and constraints have been incorporated in the problem to increase the potential applications of our work. Some of these are: *(i)* the inclusion of both expensive and non-expensive products on each display table and horizon; *(ii)* the achievement of a minimum profit margin per table and horizon; *(iii)* the consideration of dynamic (novelty-based) and correlated (combination-based) attractiveness levels; and *(iv)* the consideration of dynamic selling prices.

As solution approaches, a biased-randomized GRASP and a biased-randomized ILS have been proposed. To test these methodologies, a complete set of instances was generated by considering realistic assumptions and different design factors. In our approach, it is assumed that the attractiveness value of each product can be estimated using historical data obtained from an omnichannel environment. The experimental results show that both biased-randomized methodologies are able to provide, in short computing times, solutions that clearly outperform the human-behavior and other more standard methodologies. Additionally, a numerical study has shown that our biased-randomized algorithms are very competitive when compared with non-linear solver engines, obtaining better or similar solutions in much shorter computing times.

By increasing the attractiveness level of retail display tables in a short time horizon, managers can reduce customer attrition and, as a consequence, increase sales revenue in their stores. Using biased-randomized algorithms to maximize the attractiveness of products assigned to display tables in the considered scenario represents a clear enhancement over current practice, which might typically require many hours of a dedicated expert to generate even a feasible solution.

Table 5.6: Comparison of the results obtained by the proposed methodologies.

Instance	Best Solution (attractiveness)						AVG Attractiveness (SE)						GAP (%) w.r.t. BKS						AVG Time (sec.)												
	G		BR-GR		GR		ILS		BR-ILS		G		BR-GR		GR		ILS		BR-ILS		G		BR-GR		GR		ILS		BR-ILS		
	G	BR-GR	GR	ILS	BR-ILS	G	BR-GR	GR	ILS	BR-ILS	G	BR-GR	GR	ILS	BR-ILS	G	BR-GR	GR	ILS	BR-ILS	G	BR-GR	GR	ILS	BR-ILS	G	BR-GR	GR	ILS	BR-ILS	
a500m5i1	100,167	104,906	104,748	103,981	104,422	100,167	(0)	104,497	(73)	104,440	(51)	103,482	(82)	104,104	(66)	4.52	0.00	0.15	0.88	0.46	0	8	8	8	8	8	8	8	8	8	10
a500m10i1	188,711	196,372	195,804	194,621	195,815	188,711	(0)	196,021	(78)	195,541	(78)	193,967	(108)	195,370	(78)	3.90	0.00	0.29	0.89	0.28	0	15	17	15	15	15	15	15	15	20	
a1000m5i1	110,916	113,875	113,097	113,223	113,515	110,916	(0)	113,630	(53)	112,708	(48)	112,744	(79)	113,292	(54)	2.60	0.00	0.68	0.57	0.32	0	15	14	17	17	17	17	17	21	21	
a1000m10i1	212,437	217,987	215,415	216,246	217,287	212,437	(0)	217,540	(111)	214,911	(100)	215,857	(95)	217,058	(60)	2.55	0.00	1.18	0.80	0.32	0	30	30	36	36	36	36	40	40		
a1500m5i1	114,397	117,609	115,983	116,460	117,437	114,397	(0)	117,289	(61)	115,750	(47)	116,112	(72)	117,057	(58)	2.73	0.00	1.38	0.98	0.15	0	23	23	41	41	41	41	36	36		
a1500m10i1	223,202	228,676	223,475	226,916	227,794	223,202	(0)	227,827	(159)	223,053	(75)	226,517	(81)	227,418	(68)	2.39	0.00	2.27	0.77	0.39	0	48	44	69	69	69	69	66	66		
a2000m5i1	117,377	120,696	118,292	119,967	120,628	117,377	(0)	120,545	(41)	118,147	(43)	119,599	(59)	120,442	(42)	2.75	0.00	1.99	0.60	0.06	0	33	32	40	40	40	40	30	30		
a2000m10i1	231,988	236,010	229,241	234,086	235,096	231,988	(0)	235,431	(111)	229,029	(60)	233,674	(89)	234,831	(64)	1.70	0.00	2.87	0.82	0.39	0	60	57	65	65	65	65	63	63		
AVG	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	2.89	0.00	1.35	0.79	0.29	0	29	28	37	37	37	37	36	36		

Chapter 6

Applications in Production

This chapter ¹ studies the combination of a Hybrid Flow-Shop with a Vehicle Routing Problem. In this integrated system, the production and distribution phases are considered conjointly. In other words, a set of jobs (or products) are processed at the production stage. As soon they are concluded, the finished products are grouped into batches, which are delivered as soon they are loaded into the cargo vehicle. To solve this problem, a biased-randomized variable neighborhood descent (BR-VND) metaheuristic is proposed. This approach combines the use of constructive heuristics for coping with the production stage, i.e., the hybrid flow-shop, while the second stage, which regards the last-mile delivery, is solved through a savings-based heuristic.

6.1 The Hybrid Flow-Shop Vehicle Routing Problem

In most supply chains, there is an increasing need to coordinate efforts of suppliers, producers, and carriers to efficiently deliver products to customers so that waste and lead times are reduced. The production and distributions phases are critical in any supply chain: finished products are transferred from production centers to warehouses or distribution centers by cargo vehicles. In order to enhance the operational performance, both phases need to be considered while optimizing operations. Still, due to the complexity of these phases, traditional approaches usually consider them as two isolated and independent problems (Chen, 2010).

In this Section, a more holistic approach is offered by considering the production and distribution phases conjointly. This is the case, for example, of distributing medical tests or vaccines to local health centers –so they can be administrated to the population as soon as possible– while these items are being produced, in large quantities, at a central laboratory. Hence, the production phase can be modeled as a hybrid flow-shop (HFS) environment, while the distribution phase can be modeled as a vehicle routing problem (VRP). Accordingly, the combined problem can be referred to as a hybrid flow-shop vehicle routing problem (HFS-VRP). Figure 6.1 depicts this combined problem, where the jobs are processed

¹The contents of this chapter are based on the following work:

- **Martins, L. C.**; Gonzalez, E.; Hatami, S.; Juan, A. A.; Montoya, J. (2021): [Combining Production and Distribution in Supply Chains: the Hybrid Flow-Shop Vehicle Routing Problem](#). *Computers & Industrial Engineering*, 159, 107486.

at the production stage, finished products are grouped into batches, and these batches are delivered as soon they are loaded into the cargo vehicle. By grouping finished items into batches, their delivery to customers can be performed while the production system is manufacturing new ones. A single capacitated cargo vehicle is available to perform the deliveries, which means that several trips must be necessary to deliver all products to consumers.

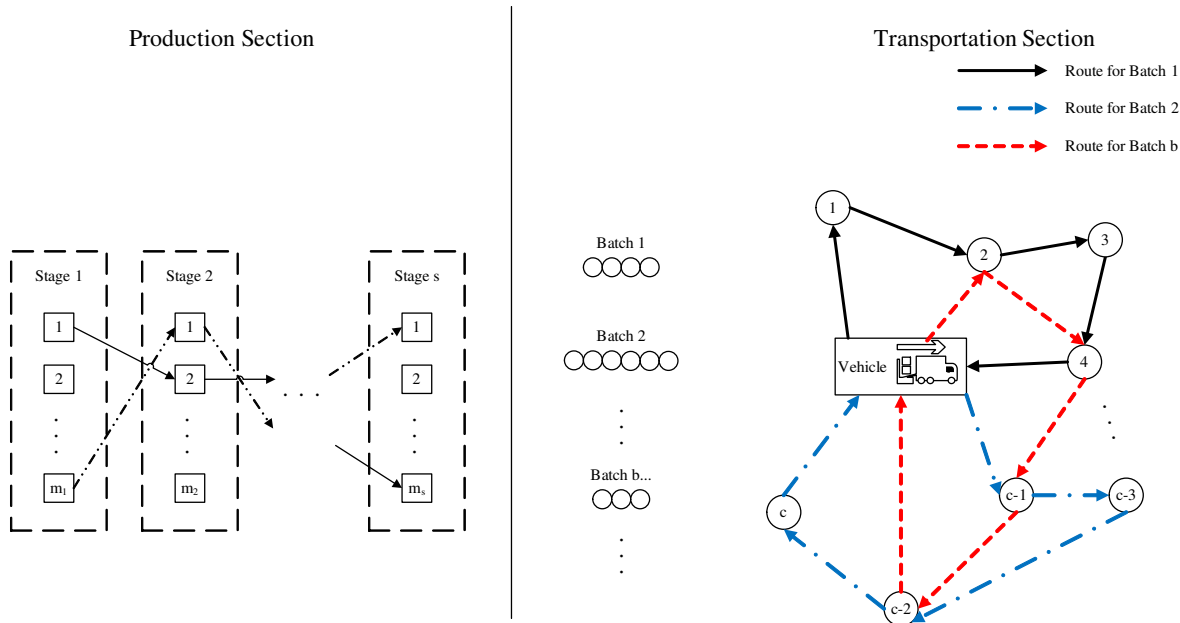


Figure 6.1: Combined production and distribution operations.

The goal when solving the HFS-VRP is to minimize the total time elapsed since the start of the manufacturing process and the delivery of the last customer's demand. Accordingly, three different and interrelated decisions have to be made: (i) determining the job sequence on each machine at the production phase; (ii) assigning the finished jobs to a proper batch for deliver; and (iii) determining adequate routes for vehicles in order to deliver jobs to customers. To the best of our knowledge, and despite its many applications in supply chain management, this is the first time that such a combined hybrid flow-shop and vehicle routing problem has been discussed in the scientific literature.

To cope with the complexity of the HFS-VRP, a biased-randomized variable neighborhood descend (BR-VND) metaheuristic is proposed. Additionally, a new set of instances, which are based on some well-known benchmark instances of both the HFS and the VRP, are introduced. In order to validate the performance of the BR-VND, we first solve the HFS and compare our results with the ones existing in the literature. Then, we extend the proposed metaheuristic to solve the HFS-VRP by considering the new set of instances. Finally, the performance of the proposed constructive heuristics and metaheuristic are evaluated.

6.1.1 Literature Review

The analysis of combined production and distribution processes has been quite common from a tactical and strategical points of view. Hence, many review papers have been published on these areas, e.g.: Thomas and Griffin (1996), Cohen and Mallik (1997), Vidal and

Goetschalckx (1997), Erengüç et al. (1999), Sarmiento and Nagi (1999), Goetschalckx et al. (2002), Chen (2004), Meixell and Gargeya (2005), Saenz et al. (2015) and Koç et al. (2017). However, research at the operational level is much more recent and scarce, with just a few articles discussing the combination of production scheduling and vehicle routing operations (Chen, 2010).

According to Karaođlan and Kesen (2017), the integrated production scheduling and transportation problem can be classified into three categories, depending on the method employed for sorting the deliveries. The first category consists of simple methods like direct shipping, without a routing process: an order, a batch to a single client, or a batch to multiple customers delivered as soon as the production process is finished. Examples of the first category can be found in the works of Cakici et al. (2014) and Wang et al. (2016). The second category involves fixed transportation departure dates with a predetermined departure time for each vehicle, e.g.: Stecke and Zhao (2007) and Hajiaghaei-Keshteli et al. (2014). The third category consists of vehicle routing decisions to be made, involving the determination of departure times. A complete discussion on the integrated production scheduling and distribution operations can be found in Chen and Vairaktarakis (2005), Wang et al. (2015), and Moons et al. (2017). Our review will mainly focus on the third category, which is also the less studied one in the literature (Karaođlan and Kesen, 2017).

Li et al. (2005) considered applications where one manufacturing factory and one delivery process are studied. Two objectives were analyzed: the customer service level and total distribution costs. Customer service was studied with two different measures: mean completion time and makespan. Dispatching costs included fixed and variable costs, the latter depending on the traveled distance. The authors proposed several mathematical models and, when the problems could not be solved exactly, heuristic approaches were proposed to obtain near-optimal solutions. Li and Vairaktarakis (2007) solved a bundling operations problem in which two dedicated machines perform two different tasks of the same job that can be executed in parallel. The job is finished when the two tasks are completed. Then, transportation is carried out by various vehicles. Decisions to be made are the sequencing of jobs into machines, the number of vehicles for transportation, and the routes they have to follow. The objective was to minimize the total cost of transportation and the waiting cost of customers. The authors proposed a polynomial-time algorithm and several heuristics to solve the problem. Armstrong et al. (2008) considered a single-machine problem in which jobs belonging to the same production order must be processed one after the other. Production orders had time windows for delivery, with no inventory allowed between the production and the transportation stages.

Armstrong et al. (2008), Geismar et al. (2008) and Geismar et al. (2011) considered the production and distribution of perishable products, which require avoiding waiting times before delivery. Armstrong et al. (2008) have included delivery time windows specified by the customer. Due to the limited resources of production and delivery processes, the whole demand cannot be met. Thus, the decision is to select the subset of customers that can be served, such in a way that the total satisfied demand is maximized. To solve this problem, the authors proposed a branch-and-bound algorithm. Geismar et al. (2008) have

included the vehicle routing problem in the decision process. Since this is an *NP-hard* problem, these authors developed lower bounds that were used by a two-phase metaheuristic algorithm. The first phase is a GA that provides a local optimum sequence for completing the products of the selected customers. The second phase divides the sequence into various subsets and uses the Gilmore-Gomory algorithm (Gilmore and Gomory, 1961) to order the sub-sequences. Geismar et al. (2011) studied the same problem but considering intermediate hubs, which cluster some customers. The objective here was to minimize the total cost of production and transportation operations while respecting the product lifetime and delivery capacity of vehicles.

Farahani et al. (2012) solved a cost minimization problem in the production and distribution scheduling of catering foods by employing an iterative hierarchical approach. In the first stage, the authors applied an aggregation procedure to create batches of orders with similar characteristics. Then, a block planning scheme is proposed to schedule the batches. Next, a heuristic is used to solve the delivery problem. Finally, the iterative approach is implemented to coordinate both schedules. Condotta et al. (2013) considers a single machine in the production stage and a given fleet of vehicles with limited capacity to deliver final products. Jobs have a due date for delivery, and the goal is to minimize the lateness. A TS algorithm was proposed for obtaining partial solutions at the production stage. Later, the TS was hybridized with an optimal transportation schedule. Hajiaghahi-Keshteli and Aminnayeri (2014) proposed one heuristic procedure and two metaheuristics, GA and SA, to maximize customer service at the minimum total cost. Their GA obtained the best results, especially as the instance size increases.

Low et al. (2014) considered the production of a variety of products associated with one customer as a batch. The batches might be delivered immediately after completion, or might be grouped with other batches for delivering to the corresponding retailers. A heterogeneous fleet of vehicles was considered to minimize total costs. They proposed a MILP model and two GAs. Kang et al. (2016) solved a real case from the semiconductor industry. Constraints, such as job clusters, production costs depending on the job clusters, setup costs, and transportation costs of multiple vehicles were considered. The authors proposed a MILP model and a GA to minimize the total cost for large instances.

Karaođlan and Kesen (2017) proposed a branch-and-cut algorithm to minimize the makespan in the production of a single product with limited shelf life. For delivery purposes, there is only one single vehicle with limited capacity. Fu et al. (2017) analyzed the problem with unrelated parallel machines and job splitting during the production stage. The transportation stage included delivery time windows and the delivery of jobs in batches using heterogeneous vehicles. Two objectives were evaluated with the use of an iterative heuristic: the setup costs minimization at the production stage and transportation costs for delivery.

6.1.2 Problem Definition

As introduced, the HFS-VRP combines the HFS with a VRP. In the production phase, a set J of jobs (items) must be processed. Each job has to go through a set S of sequential stages. At each stage $s \in S$, a set M_s of parallel and identical machines are available to process the

job. Given a job $j \in J$, its processing time, in stage $s \in S$, is given by $p_{js} > 0$. Regarding the distribution phase, a set C of customers and one vehicle that makes multiple trips are considered. On each trip, the vehicle delivers a batch of jobs. Each job $j \in J$ belongs to a specific customer $c \in C$ and requires a loading capacity of $q_j > 0$, being $Q \gg \max_{i \in J} \{q_j\}$ the maximum loading capacity of the vehicle.

In order to speed up the delivery process, finished items are grouped into batches that can be delivered to customers while the production system is manufacturing new ones. Since the loading capacity of the cargo vehicle is limited, multiple trips might be necessary to perform all the required deliveries. The vehicle trip starts at the production plant and meets a sequence of customers. After meeting and delivering the jobs to the relevant customers, the vehicle should return to the production plant to pick up and distribute the remaining jobs.

The transportation phase can be considered as a directed graph $G = (V, A)$, where $V = \{0, 1, \dots, c\}$ is composed by a single depot (0), and c customers, and the set $A = \{(x, y) : x, y \in C', x \neq y\}$ represents the arcs connecting pairs of nodes. Both the production plant and vehicle are located at the depot. Transportation times between each pair of nodes are considered symmetric. Accordingly, the travel time associated with traveling from node i to node j is denoted by $t_{ij} \geq 0$. In this context, the goal is to minimize the total time elapsed since the start of the manufacturing process and the delivery of the last customer's demand, i.e., the makespan of the hybrid problem. In another point of the view, the considered objective is to minimize the makespan of the production section plus the transportation time of the jobs from the production section to corresponding customers. This value is not simply equal to the summation of the makespan (the overall completion time of jobs belong to the last customer) and the transportation cost to the last customer. Sometimes, this value might be greater than the aforementioned summation. Since the vehicle must return to the production plant (depot) after delivering the current batch, possibly, its arrival time to the depot might be greater than the completion time of the batch that should be distributed in the next turn. Therefore, in this case, the solution cost is the maximum value between the completion time of batch including the jobs ready to distribute and the time that the vehicle returns to the depot, plus the transportation cost of the ready batch from depot to the relevant customer(s).

In Appendix B.2 (Section B.2.2), a MILP model for the HFS-VRP is introduced. Moreover, considering the NP-hardness of the problem, a first lower bound (LB) calculation for the problem is addressed, in order to evaluate the performance of our proposed algorithms. Both the results obtained when solving the MILP model and the LB values are considered in the computational experiments and results section (Section 6.1.7).

As mentioned before, the following three decisions have to be made in order to solve HFS-VRP problem: (i) determining the job sequence at the production stage; (ii) assigning the finished jobs to a proper batch for delivery; and (iii) defining the routing plan for the single cargo vehicle. In order to give a better understanding of the problem, Figure 6.2 provides a numerical example with 6 jobs ($n = 6$) and 3 stages ($s = 3$), in which the first and third stages are composed of 3 machines each ($m_1 = 3$ and $m_3 = 3$), while the second stage

is composed of a single machine ($m_2 = 1$).

1. At the HFS stage, each job is described by a tuple (j, c_j, q_j) , in which j is the job identifier, c_j is the customer who requires the job j , and q_j is the loading volume of job j . For instance, in tuple $(1, 1, 10)$, the job 1 is requested by customer 1, and consists of 10 demand volume units. Once processed in the first stage –with a completion time of 100 time units– the job 1 can be processed in the following stage from time 100, and so on. The remaining jobs follow the same interpretation.
2. The second stage aims to join processed jobs into batches that meets the capacity constraint of the cargo vehicle. Each batch corresponds to one trip. In this example, the vehicle has a capacity of 50 demands units. A batch b are represented by the set of jobs and the tuple (CR_b, TD_b) , where CR_b and TD_b represent the completion time and total volume of batch b , respectively. For example, the batch 1 is composed of jobs 3 and 2, has a completion time is 600 time units, and its total volume is 45 units.
3. The last stage regards the vehicle routing process. For the first batch 1, the vehicle starts its delivery at time SR_1 , i.e., 600 time units. In this stage, each node is characterized by the tuple $[TV_{c,b}, RD_{c,b}]$, in which $TV_{c,b}$ represents the arrival time at node c of batch (trip) b , and $RD_{c,b}$ represents the remaining loaded demand at node c of batch (trip) b . For instance, the vehicle arrives at the depot after delivering the jobs at time 820 with no loaded demand. For the next route, the delivery starts at time 820, since the vehicle arrives at the depot after batch 2 being ready for delivery at time 800, i.e., the $\max(800, 820)$. In case the vehicle is ready for delivery before the conclusion time of the batch, it must wait for the time needed for the batch to be ready and loaded. The same is done for the remaining batches.
4. Finally, the solution cost is given by the time in which the vehicle returns to the depot after delivering the jobs from the last batch. In this example, the integrated cost is 1210.

6.1.3 Solution Method: From a Heuristic to a Metaheuristic

Since the HFS and the VRP are both *NP-hard* problems (Ruiz and Vázquez-Rodríguez, 2010; Lenstra and Kan, 1981a), so it is the composed HFS-VRP. Therefore, the use of metaheuristic approaches becomes necessary to solve large-sized instances in reasonable computing times. Hence, we propose an algorithm that combines biased-randomization (BR) techniques (Gonzalez-Martin et al., 2012) with the well-known variable neighborhood descent (VND) framework. The latter is a variant of the VNS metaheuristic framework (Mladenović and Hansen, 1997). The VNS is an enhanced local search strategy that systematically explores the solution space by changing the neighborhood structure. The local optimum provided by one neighborhood structure is not necessarily the same as the one provided by another neighborhood structure. In this way, the search becomes more flexible by exploring different neighborhood structures (Burke et al., 2008). The VND starts by employing

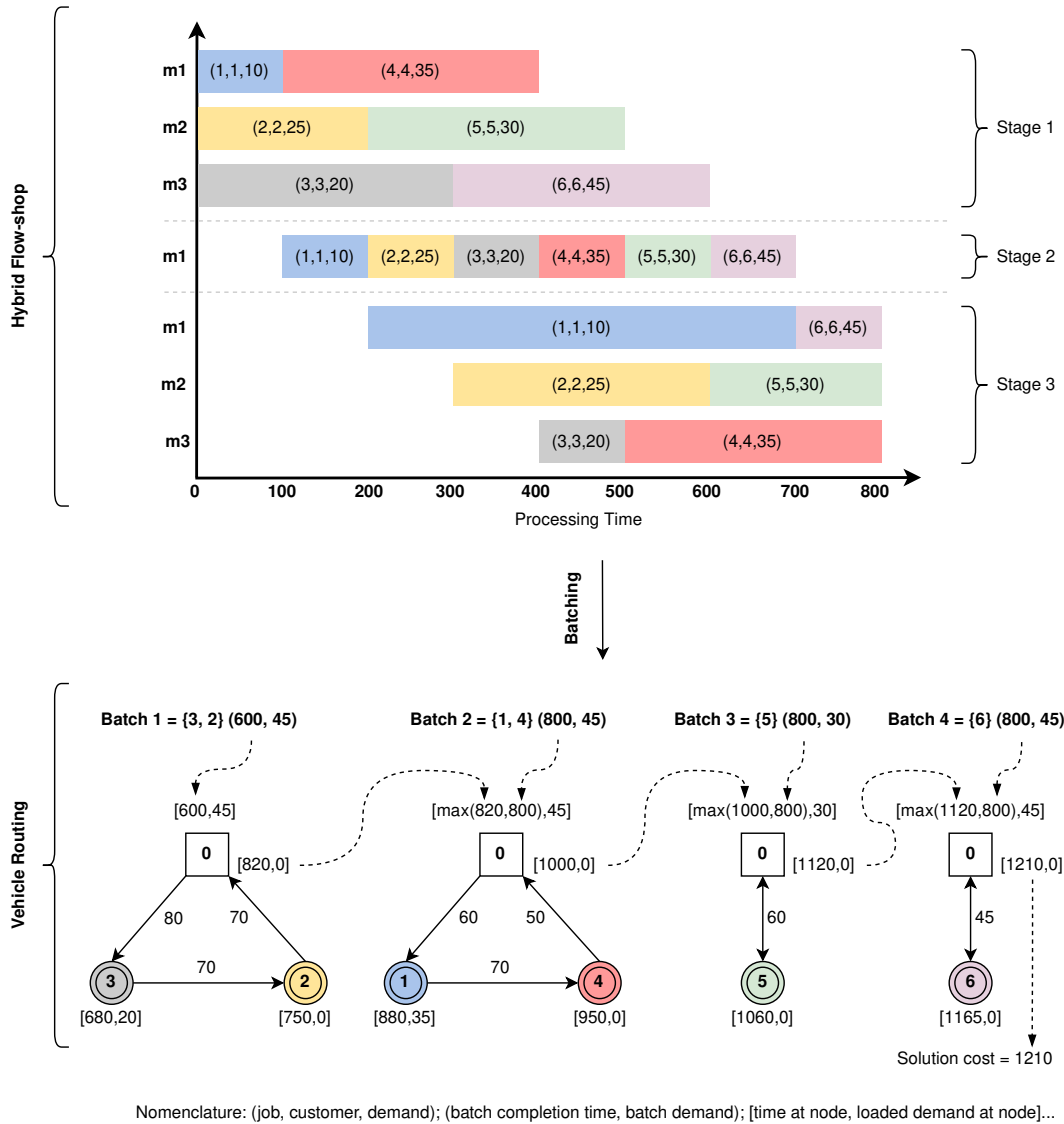


Figure 6.2: Combined production and distribution operations.

an initial structure N_1 . The searching process continues until no further improvement is reached. Then, a new neighborhood structure, N_2 , is explored. If a new local optimum is obtained, the VND returns back and starts again with N_1 . Otherwise, it continues with the next neighborhood structure, N_3 . This process goes on until the last neighborhood structure is reached. In our biased-randomized variable neighborhood descent (BR-VND) algorithm, we first create an initial solution, which is then iteratively improved by employing a set of neighborhood structures (Figure 6.1). More details on our algorithm are provided in the following subsections.

6.1.4 Solution Representation and Loading Strategy

We consider two different solution representations. The first one, SR_1 , is a complete sequence (permutation) of all jobs, and does not make any assumption about the the assignment of jobs to customers. Hence, using this representation it is possible to consider $n!$

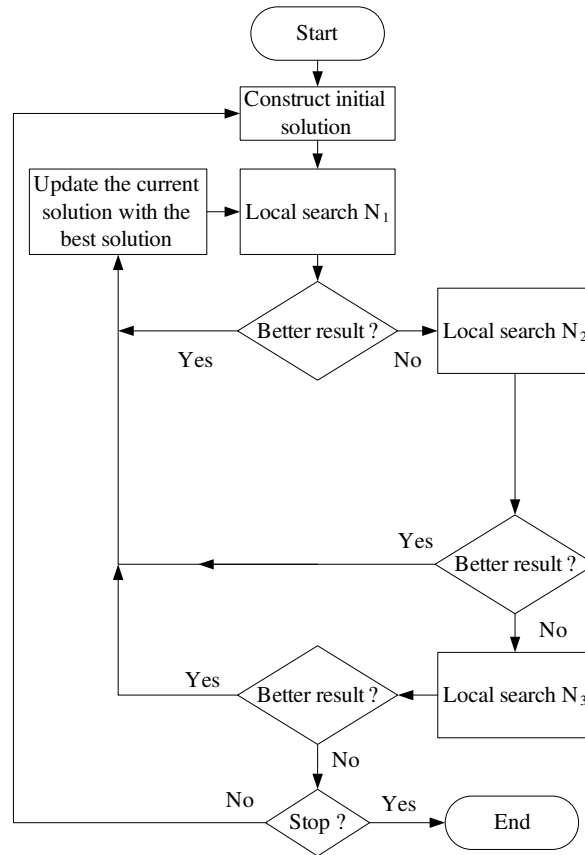


Figure 6.3: Flow-chart of the BR-VND algorithm.

different permutations. The second solution representation, SR_2 , also employs a sequence of jobs. This time, however, jobs belonging to the same customer appear together in the sequence.

The BR-VND starts by generating an initial solution. We consider biased-randomized versions of the following constructive heuristics to generate this initial solution: the NEH heuristic (BR-NEH); the short processing time bottleneck heuristic (BR-SPTB); and the backward largest processing time bottleneck heuristic (BR-bLPTB). Each constructive heuristic is applied to solution representations SR_1 and SR_2 . In order to load batches of jobs on a vehicle with limited capacity, we consider two different vehicle loading strategies. In the first one, VLS_1 , jobs are loaded by increasing order of completion times. Let us consider, for example, a single vehicle with a maximum capacity of 50 unit per trip, and 5 jobs (j_1, j_2, \dots, j_5) with the following completion times (second element in the list) and volume capacities (third element in the list): $\{j_1, 38, 15\}$, $\{j_2, 24, 35\}$, $\{j_3, 31, 5\}$, $\{j_4, 20, 10\}$, and $\{j_5, 15, 30\}$. Thus, VLS_1 will determine the following loading plan of jobs: $\{j_5, j_4\}$ in trip 1, $\{j_2, j_3\}$ in trip 2, and $\{j_1\}$ in trip 3. In the second loading strategy, VLS_2 , the goal is to load the maximum possible volume in each trip. Hence, when applying this second loading strategy to the previous numerical example, the loading plan will be as follows: $\{j_5, j_4, j_3\}$ in trip 1, $\{j_2\}$ in trip 2, and $\{j_1\}$ in trip 3. In order to investigate the effects of different initial solutions ($IniSol$) and loading strategies (VLS), we design twelve variants of the algorithm. These variants employ the same solution representation and have the same neighborhood structures, but they use

different initial solutions and loading strategies.

6.1.5 Generating an Initial Solution

For the generation of the initial solution (*IniSol*), we propose the implementation of simple dispatching rules, such as the shortest processing time (SPT) and longest processing time (LPT) ones, which are adapted to the HFS problem. Both the SPT and the LPT generate job permutations that are based on sorting the total processing time of jobs according to an ascending and a descending order, respectively. Once all operations of one job are completed on the production phase, the job can be delivered to its demanding customer. Therefore, a vehicle is loaded and routed. For the routing process, we employ a biased-randomized version of the popular savings heuristic (Quintero-Araujo et al., 2017). The biased-randomization processes employed in this section, both during the scheduling and the routing phases, make use of Geometric probability distributions, as proposed in (Ferrer et al., 2016) for the scheduling and in Gonzalez-Martin et al. (2018) for the routing.

6.1.5.1 The BR-NEH Heuristic

The first and second initial solutions are generated through the biased-randomized version of the NEH heuristic (Nawaz et al., 1983). The extension of the NEH to the HFS problem provides good results (Naderi et al., 2010). In the first initial solution, *BR – NEH1*, the biased-randomization process is applied to a job sequence β in order to generate a job sequence π . Then, a ‘shift-to-left’ operator (Juan et al., 2014) is used to improve the current solution. In the second initial solution, *BR – NEH2*, the NEH heuristic and a biased-randomization processes are jointly applied to all jobs belonging to each customer $c \in C$. Hence, a partial job sequence π_c is constructed for each customer $c \in C$ via the biased-randomized NEH heuristic. Next, the list of customers is sorted by ascending order of their jobs’ makespan. A complete job sequence, π_T , is obtained by putting together all the partial job sequences.

6.1.5.2 The BR-SPTB Heuristic

The idea of the third and fourth initial solutions make use of a biased-randomized version of the short processing time bottleneck heuristic proposed by Pan et al. (2014). In many cases, bottlenecks in a system are generated by a single component (Liao et al., 2012). As Paternina-Arboleda et al. (2008) mentioned, a stage is a bottleneck when it has the largest flow ratio between the workload and the total available capacity. The SPTB heuristic sorts jobs by their total processing time, from the first stage to the bottleneck one. To generate the third initial solution, *BR-SPTB1*, the biased-randomized process is applied to the job sequence. The fourth initial solution, *BR-SPTB2*, is similar to the second one –it also works separately with the jobs that belong to each customer. The only difference is that in *BR-SPTB2* the partial job sequence for each customer is obtained via a biased-randomized version of the SPTB heuristic.

6.1.5.3 The BR-bLPTB Heuristic

The last two initial solutions are the *BR-bLPTB1* and *BR-bLPTB2*, which make use of biased-randomized version of the bLPTB heuristic. In the *BR-bLPTB1*, the jobs are sorted by their total processing time, in descending order, from the bottleneck stage to the last stage. In the *BR-bLPTB2*, the biased-randomized heuristic is applied to jobs belonging to each customer.

6.1.6 Neighborhood Structures

Our proposed BR-VND algorithm employs three neighborhood structures for the first solution representation, SR_1 , and two for the second solution representation, SR_2 .

6.1.6.1 The SR_1 Neighborhood Structures

Pseudo-codes 17 to 19 show the three proposed neighborhood structures. The first neighborhood for SR_1 is referred to as LS_{C1} and attempts to improve the objective function by examining different complete job sequences. LS_{C1} provides a list of complete job sequences by removing a single job from π_T and inserting it into all possible $n - 1$ positions of π_T . The newly created job sequences are evaluated by assigning the jobs to the machines on the stages. If a new sequence provides a better objective function, then π_T is updated and all jobs are reinserted again. Otherwise, the search continues with the next job. The second neighborhood for this solution representation, LS_{P1} , works with partial job sequences: given a machine g in a stage k , it takes all jobs in π_{kg} and inserts them, considering all possible positions, both in the same as well as in any other machines at the stage k . When all jobs in machine g are considered, the search is continued with the next machine in stage k and, once these have been covered, with the machines in the next stage. The last neighborhood for SR_1 , LS_{CS1} , is similar to LS_{C1} . It works over the jobs on a complete job sequence, π_T . The complete job sequence for each stage k , $\pi_{T(k)}$ is constructed. It extracts and reinserts each job into all possible $n - 1$ positions of $\pi_{T(k)}$.

Pseudocode 17: Neighborhood structures, LS_{C1} .

```

1  $l \leftarrow 1$ 
2 while  $l \leq n$  do
3   Remove job  $a$  located at position  $l$  of  $\pi_T$ 
4   Insert  $a$  into all  $n - 1$  possible positions of  $\pi_T$ 
5   Evaluate all obtained  $\pi_T$  by assigning jobs to machines of the stages
6   if a better objective function is obtained then
7     | update  $\pi_T$ 
8   else
9     |  $l \leftarrow l + 1$ 
10  end
11 end

```

6.1.6.2 The SR_2 Neighborhood Structures

Pseudocode 18: Neighborhood structures, LS_{P1} .

```

1  $k \leftarrow 1$ 
2 while  $r \leq s$  do
3    $g = 1, w \in M_k, w \neq g$ 
4   while  $g \leq m_k$  do
5      $j \leftarrow 1$ 
6     while  $j \leq |N(\pi_{kg})|$  do
7       Remove job  $a$  located at position  $j$  of  $\pi_{kg}$ 
8       Insert  $a$  into all possible positions of current  $\pi_{kg}$  and other  $\rho_{kw}$ 
9       if a better objective function is obtained then
10        | update  $\pi_{kg}$  and other  $\pi_{kw}$ 
11        | else
12        |    $j \leftarrow j + 1$ 
13        | end
14      end
15       $g \leftarrow g + 1$ 
16    end
17     $k \leftarrow k + 1$ 
18 end

```

Pseudocode 19: Neighborhood structures, LS_{CS1} .

```

1  $k = 1, t \in s, t \geq k$ 
2 while  $k \leq s$  do
3    $l = 1$ 
4   while  $l \leq n$  do
5     Obtain complete job sequence for stage  $k$ ,  $\pi_{T(k)}$ 
6     Remove job  $a$  located at position  $l$  of  $\pi_{T(k)}$ 
7     Insert  $a$  into all  $n - 1$  possible positions of  $\pi_{T(k)}$ 
8     Evaluate all obtained  $\pi_{T(k)}$  by assigning jobs to machines of the stages  $t$ 
9     if a better objective function is obtained then
10      | update partial job sequences on machines at stage  $k$  and stages  $t$ 
11      | else
12      |    $l = l + 1$ 
13      | end
14    end
15     $k \leftarrow k + 1$ 
16 end

```

The first neighborhood proposed for the SR_2 solution representation, LS_{CS2} , works over the jobs that belong to the same customer: given an entire sequence π_T , it extracts all jobs associated with each customer as a block, and insert this block in all possible positions of π_T . The second neighborhood designed for SR_2 , LS_{C2} , is similar to LS_{C1} and works over the jobs on the complete job sequence π_T . However, in the case of LS_{C2} , all jobs belonging to a customer $c \in C$ are extracted and inserted into all possible positions of the partial sequence associated with c .

6.1.7 Computational Experiments and Results

As mentioned before, this is the first time that a combined hybrid flow-shop and vehicle routing problem is discussed in the scientific literature. Consequently, no benchmark instances are available in the literature to experimentally evaluate the proposed solution approaches. In this way, a new set of instances is introduced. Moreover, the proposed algorithm was developed in Java and the tests were performed on an Intel Core i7-8550U processor with 16 GB of RAM.

6.1.7.1 Generation of Instances

The proposed instances are based on some well-known benchmark instances of both the HFS and the vehicle routing problems. Accordingly, the four instance factors listed in Table 6.1 were considered in this process.

Table 6.1: Instance factor for the small and large instances.

Instance factor	Symbol	Instance type			
		Small		Large	
		Number of levels	Values	Number of levels	Values
Number of jobs	n	3	6, 8, 10	3	60, 80, 100
Number of stage	s	3	2, 3, 4	3	5, 8, 10
Number of customer	c	3	2, 3, 4	8	16, 19, 20, 22, 23, 31, 32, 33
Vehicle loading capacity	v	2	20, 30	2	100, 200

The number of identical parallel machines at each stage $k \in S$, m_k is generated using a uniform distribution $U[1, 5]$. Processing times of jobs on the HFS section are fixed to be integer values from a uniform distribution $U[1, 99]$, as commonly defined in the scheduling literature. The volume capacity of each job $j \in N$, l_j is uniformly generated in the range of $U[5, 10]$ for small instances and $U[10, 30]$ for large instances. Since the customers should place in a certain geographic location, we have used some well-known set of benchmarks in the VRP literature. Eight VRP instances have been selected from a set of instances A, B, E and P , available at <http://vrp.atd-lab.inf.puc-rio.br/index.php/en/>. Regarding the number of the jobs, we have selected some acceptable instances where $n > c$. These instances are different among their scattered or clustered topology. In the case of small instances, the customer data was taken from the first customers of VRP mentioned instances. In particular, for small instances type, one test instance was generated for each combination of n, s, c , and v , obtaining a total of 54 small-sized instances. On the other hand, for large-sized instances, five test instances were created for each combination of n, s, c , and v , leading to a total of 720 large instances.

6.1.7.2 Results of Small Instances

The MILP model presented in Appendix B.2 (Section B.2.2) was implemented in GLPK language with stopping criteria of 3600 seconds. Table 6.2 shows the results of the makespan of the best integer solution found after 3600s of running. As we can notice, for 75.92% of the small problem instances, i.e., 41 of the 54, no integer solution was found after one hour

of execution. From the 13 instances in which an integer solution could be found, 9 of them were optimal. The table also presents the results of our proposed lower bound ($LB_{HFS-VRP}$), the percentage that the proposed LB is below the optimal value (LB_{Dev}) (6.1), the minimum value found by our BR-VND algorithm (Min_{BR-VND}), the minimum GAP of the BR-VND in comparison with the MILP model ($GAP_{BRVND_{MILP}}$) (6.2), and the minimum GAP of the BR-VND in comparison with the proposed lower bound ($GAP_{BRVND_{LB}}$) (6.3).

$$LB_{Dev} = \frac{LowerBound_{sol} - MILP_{sol}}{MILP_{sol}} \cdot 100\% \quad (6.1)$$

$$GAP_{BR-VND_{MILP}} = \frac{BRVND_{sol} - MILP_{sol}}{MILP_{sol}} \cdot 100\% \quad (6.2)$$

$$GAP_{BR-VND_{LB}} = \frac{BRVND_{sol} - LB_{HFS-VRP_{sol}}}{LB_{HFS-VRP_{sol}}} \cdot 100\% \quad (6.3)$$

In the 13 instances with integer solution found by the MILP model, the $LB_{HFS-VRP}$ is 32.83%, on average, below the MILP result and the $GAP_{BR-VND_{MILP}}$ is 8.22%, on average. Specifically, for the problem instance $n = 6, s = 2, v = 30, c = 2$, the BR-VND found the optimal solution, and for the problem instance $n = 6, s = 3, v = 20, c = 4$, the BR-VND found a better solution than the best integer solution reached by the MILP model after an hour of execution.

For each of the 54 problem instances presented in Table 6.2, it is shown the minimum value found by our BR-VND algorithm (Min_{BR-VND}), resulted from the twelve different algorithm combinations. Since each instance is executed five times for each proposed algorithm, 3240 executions have been performed. The CPU times are not reported as they are so small. As a matter of fact, among the 3240 observed CPU times in the results, the maximum reported is 1.5 seconds. The average observed CPU time in all results is only 0.06 seconds.

n	s	v	c	MILP	Time(s)	$LB_{HFS-VRP}$	LB_{Dev}	Min_{BRVND}	$GAP_{BRVND_{MILP}}$	$GAP_{BRVND_{LB}}$
2	20	2	2	282.44*	12.1	176.22	-37.61	296.29	4.90	68.14
2	20	3	3	396.02*	81.3	239.66	-39.48	416.63	5.21	73.84
2	20	4	4	370.40*	155.3	202.66	-45.29	442.3	19.41	118.25
2	30	2	2	451.07*	65.6	437.22	-3.07	451.07	0.00	3.17
2	30	3	3	272.81*	112.3	179.66	-34.15	291.9	7.00	62.48
2	30	4	4	311.41	3600.0	206.00	-33.85	347.18	11.49	68.54
3	20	2	-	-	3600.0	348.10	-	433.65	-	24.58
3	20	3	-	-	3600.0	381.22	-	468.5	-	22.9
6	3	20	4	492.88	3600.0	276.22	-43.96	486.41	-1.31	76.1
3	30	2	-	-	3600.0	303.22	-	542.07	-	78.77
3	30	3	3	294.24*	12.0	182.00	-38.15	357.77	21.59	96.58
3	30	4	4	421.28	3600.0	372.22	-11.65	424.03	0.65	13.92
4	20	2	2	388.58*	24.7	254.22	-34.58	427.65	10.05	68.22
4	20	3	-	-	3600.0	359.22	-	481.43	-	34.02
4	20	4	-	-	3600.0	306.66	-	500.24	-	63.13
4	30	2	2	346.07*	3.6	229.00	-33.83	404.21	16.80	76.51

Continued on next page *Optimal solution

Table 6.2 – continued from previous page

n	s	v	c	MILP	Time(s)	$LB_{HFS-VRP}$	LB_{Dev}	Min_{BRVND}	$GAP_{BRVND_{MILP}}$	$GAP_{BRVND_{LB}}$
4	30	3		388.90*	89.2	222.00	-42.92	423.23	8.83	90.64
4	30	4		-	3600.0	467.00	-	537.5	-	15.1
	2	20	2	-	3600.0	282.10	-	534.22	-	89.37
	2	20	3	-	3600.0	399.54	-	534.03	-	33.66
	2	20	4	-	3600.0	301.10	-	470.02	-	56.1
	2	30	2	-	3600.0	194.66	-	385.54	-	98.06
	2	30	3	-	3600.0	337.22	-	417.13	-	23.7
	2	30	4	-	3600.0	210.66	-	482.07	-	128.84
	3	20	2	-	3600.0	312.10	-	448.73	-	43.78
	3	20	3	-	3600.0	238.66	-	450.2	-	88.64
8	3	20	4	-	3600.0	371.10	-	530.36	-	42.92
	3	30	2	-	3600.0	231.66	-	434.69	-	87.64
	3	30	3	-	3600.0	320.22	-	385.24	-	20.31
	3	30	4	-	3600.0	251.66	-	381.06	-	51.42
	4	20	2	520.51	3600.0	373.22	-28.30	532.24	2.25	42.61
	4	20	3	-	3600.0	470.22	-	584.38	-	24.28
	4	20	4	-	3600.0	353.10	-	620.77	-	75.81
	4	30	2	-	3600.0	384.22	-	473.43	-	23.22
	4	30	3	-	3600.0	317.66	-	527.48	-	66.05
	4	30	4	-	3600.0	281.22	-	426.87	-	51.79
	2	20	2	-	3600.0	382.54	-	565.22	-	47.76
	2	20	3	-	3600.0	390.54	-	632.55	-	61.97
	2	20	4	-	3600.0	407.54	-	587.16	-	44.08
	2	30	2	-	3600.0	285.22	-	632.22	-	121.66
	2	30	3	-	3600.0	324.22	-	427.24	-	31.78
	2	30	4	-	3600.0	234.22	-	433.61	-	85.13
	3	20	2	-	3600.0	567.22	-	596.07	-	5.09
	3	20	3	-	3600.0	417.54	-	553.4	-	32.54
	3	20	4	-	3600.0	533.22	-	640.28	-	20.08
	3	30	2	-	3600.0	657.22	-	663.22	-	0.91
10	3	30	3	-	3600.0	347.22	-	468.43	-	34.91
	3	30	4	-	3600.0	328.22	-	482.73	-	47.08
	4	20	2	-	3600.0	528.22	-	593.22	-	12.31
	4	20	3	-	3600.0	458.54	-	660.51	-	44.05
	4	20	4	-	3600.0	559.22	-	714.47	-	27.76
	4	30	2	-	3600.0	294.66	-	468.69	-	59.06
	4	30	3	-	3600.0	612.22	-	620.22	-	1.31
	4	30	4	-	3600.0	624.22	-	779.88	-	24.94
Average					3010.3		-32.83		8.22	51.95

*Optimal solution

Table 6.2: Results of MILP and proposed LB for small instances.

6.1.7.3 Results of Large Instances

An experimental design was carried out to test the performance of the proposed algorithms. The experiment has considered the factors n , s , v , c , $IniSol$, SR , and VLS . The levels considered for the factors n , s , v , and c were presented in Table 6.1 for the large-sized instances. The levels of $IniSol$, SR , and VLS were those presented in Table 6.3. Therefore, the treatments of the experiment were 1728, and the observations per treatment were 25. Considering that the

BR-VND is a stochastic algorithm, each one of the 720 large instances was run for five different replications. Thence, there are a total of 25 observations per treatment, since 5 instances were generated for each combination of n , s , v , and c , and each one of them was run 5 times. Therefore, a total of 43,200 replications were executed for the experiment. Each replication was limited to $40 \times n \times s$ milliseconds of running as stopping criteria. For each replication, we have calculated the GAP as $GAP = ((Algorithm_{sol} - LowerBound_{sol}) / LowerBound_{sol})$ where $Algorithm_{sol}$ is the solution obtained by a given algorithm and $LowerBound_{sol}$ is the lower bound obtained by applying the calculations presented on Appendix B.2 (Section B.2.2.2) for the corresponding instance.

Table 6.3: Test factors for instances.

Test factor	Symbol	Number of levels	Values
Initial solution	<i>IniSol</i>	3	BR-NEH, BR-SPTB, BR-bLPTB
Solution representation	<i>SR</i>	2	SR_1, SR_2
Loading strategy	<i>VLS</i>	2	VLS_1, VLS_2

Table 6.4 summarizes the results of the proposed algorithms, by comparing them with the proposed lower bound. The results are categorized by all the instance factors, i.e., n , s , v , and c . As shown in Table 6.4, from the descriptive point of view, the algorithm that combines the second solution representation SR_2 with the second loading strategy LR_2 and the initial solution BR-bLPTB is able to provide better solutions than the other ones, with a GAP of 17.91%. Besides, the algorithm with the worst performance is the one that combines SR_1 , LR_2 , and BR-SPTB, with a GAP of 21.98%. The behavior of the instance size factors, presented in Table 6.4, show that the problem becomes easier to solve when increasing the number of jobs (n), number of stages (s), and vehicle capacity (v). In the case of the number of customers (c), the best performance is presented in instances with 31 customers.

Despite the overall average GAP being 19.75%, it should be highlighted that for 48.70% of the instances, the obtained average GAP is smaller than 5%, and for 8.83% of the instances, the average GAP is between 5% and 10%.

The algorithms' CPU time consumption for large instances is summarised in Table 6.5, grouped by the instance characteristics. The algorithms that use the first solution representation (SR_1) use almost three times more CPU time than the algorithms that use the alternative solution representation (SR_2). The algorithm with BR-SPTB as the initial solution, SR_1 as solution representation, and VLS_2 as loading strategy, consumes an average of 42.18 seconds, the longest CPU time consumption compared to other algorithms. Moreover, notice how the CPU times clearly depend on the size of the instance (number of jobs n , number of stages s , and number of customers c).

In order to determine if there is a significant statistical difference among the results of Table 6.4, a multifactor Analysis of Variance (ANOVA) was also carried out. The response variable is the GAP, and the control variables are n , s , v , c , *IniSol*, *SR* and *VLS*. We tested the three assumptions of ANOVA that are normality, homoscedasticity, and independence of residuals.

Table 6.4: Average GAP over the proposed lower bound, grouped by instance characteristics.

	SR_1						SR_2					
	VLS_1			VLS_2			VLS_1			VLS_2		
	BR-NEH	BR-SPTB	BR- β LPTB	BR-NEH	BR-SPTB	BR- β LPTB	BR-NEH	BR-SPTB	BR- β LPTB	BR-NEH	BR-SPTB	BR- β LPTB
n	60	23.31	22.46	23.54	23.83	24.17	21.05	21.15	20.84	21.08	21.2	20.98
	80	20.61	19.50	20.67	21.15	21.21	17.19	17.62	17.32	16.85	17.24	17.00
	100	20.22	19.21	20.26	20.52	20.55	16.40	16.99	16.66	15.89	16.42	16.14
s	5	26.38	25.05	26.54	27.07	27.40	21.61	21.96	21.57	21.39	21.68	21.34
	8	21.82	20.94	21.9	22.18	22.24	18.95	19.31	18.97	18.63	18.99	18.73
	10	15.93	15.18	16.03	16.24	16.29	14.07	14.49	14.28	13.81	14.19	14.05
v	100	24.08	23.06	24.12	24.26	24.30	20.15	20.71	20.25	19.43	19.93	19.57
	200	18.68	17.72	18.85	19.4	19.66	16.27	16.46	16.29	16.45	16.65	16.51
c	16	13.24	12.90	13.36	13.77	13.66	11.46	11.85	11.68	11.53	11.94	11.80
	19	12.32	12.00	12.37	12.80	12.91	10.58	10.68	10.56	10.66	10.68	10.61
	20	13.86	13.46	14.15	14.33	14.37	12.18	12.38	12.21	12.23	12.34	12.12
	22	12.23	11.49	11.93	12.66	11.64	10.25	10.64	10.40	10.26	10.53	10.37
	23	48.91	46.25	49.27	49.16	46.59	41.42	42.23	41.62	40.15	40.82	40.45
	31	10.16	9.84	10.42	10.70	11.08	9.19	9.36	9.25	9.43	9.65	9.53
	32	31.68	30.04	31.93	33.41	34.24	24.76	25.50	24.7	24.77	25.75	25.07
	33	28.63	27.12	28.48	27.81	28.01	25.83	26.07	25.79	24.49	24.60	24.38
Average		21.38	20.39	21.49	21.83	21.98	18.21	18.59	18.27	17.94	18.29	18.04

Since the hypotheses of normality and homoscedasticity of samples were not fulfilled, we have performed an ANOVA-Type statistic (Brunner et al., 1997), which is a rank-based test that does not consider the assumptions of normality and homoscedasticity. According to the ANOVA-Type, all main effects are statistically significant with p -values very close to zero (lower than 0.001). Moreover, 18 of the 21 double interactions were significant. Specifically, regarding solution methods, the interaction between VLS and SR , and the interaction between $IniSol$ and SR , were statistically significant. According to the confidence intervals of rankings, with a confidence level of 95%, the best initial solution is $BR - NEH$, the best solution representation is SR_2 , and the best loading strategy is VLS_1 .

As it is known, the combination of the best levels of factors not necessarily leads to the best results. Then, the performance of the algorithm using a different combination of these factors is also studied. This combination generates twelve different algorithm configurations. In order to determine which configuration performs better, it is necessary to carry out the analysis of the triple interaction of factors *IniSol*, *SR*, and *VLS*. According to the confidence intervals of ranks for the twelve solution methods obtained from the ANOVA-Type statistic, all of the combinations that consider the *SR2* as solution representation present, statistically, the best performance. Figure 6.4 present the 95% Tukey confidence intervals for these configurations.

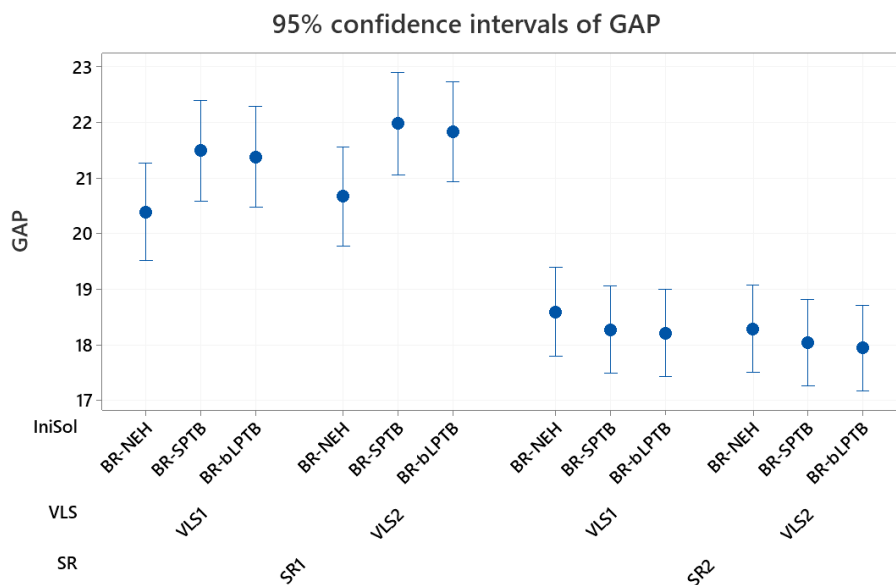


Figure 6.4: Means plot and 95% Tukey confidence intervals for different combinations of test factors

6.1.8 Conclusions

This section considered a combination of the Hybrid Flow Shop (HFS) scheduling problem with the Vehicle Routing Problem (VRP). To the best of our knowledge, this is the first time in the academic literature that this problem is approached. The problem, denoted as HFS-VRP, consists of a production section with HFS configuration to process jobs and a set of customers with a defined batch of jobs demand, followed by a distribution process where a single capacitated vehicle is available to deliver the finished batches of jobs to the final customers. The optimization objective is the minimization of the service time to the last customer, i.e., the makespan of the joint problem. As pointed out, this problem is of practical relevance, for example, to distribute medical tests or vaccines to local health centers, so they can be administrated to the population as soon as possible, while these items are being produced, in large quantities, at a central laboratory.

To solve the problem, three stages were proposed. Firstly, the MILP model. Secondly, a first lower bound of the HFS-VRP problem. Thirdly, this Section proposed a Biased-Randomized Variable Neighborhood Descent (BR-VND) metaheuristic. Twelve different

Table 6.5: CPU time (in seconds) of proposed algorithms for the large-sized instances.

	SR_1						SR_2							
	$VL\mathcal{S}_1$			$VL\mathcal{S}_2$			$VL\mathcal{S}_1$			$VL\mathcal{S}_2$				
	BR-NEH	BR-SPTB	BR-bLPTB	BR-NEH	BR-SPTB	BR-bLPTB	BR-NEH	BR-SPTB	BR-bLPTB	BR-NEH	BR-SPTB	BR-bLPTB		
n	60	15.91	17.06	16.22	17.28	15.82	18.13	15.82	7.61	8.22	7.01	8.58	8.38	7.23
	80	30.15	32.99	31.04	33.82	30.73	35.10	30.73	11.03	11.43	9.26	11.33	11.68	9.73
	100	60.99	67.90	61.01	68.41	59.73	72.89	59.73	14.03	14.35	12.23	15.15	15.00	12.34
s	5	21.03	23.71	21.52	23.84	21.00	24.99	21.00	6.80	6.80	5.75	07.09	7.30	5.46
	8	36.39	39.44	36.12	40.05	34.74	42.37	34.74	11.18	11.60	9.69	12.24	11.72	10.19
	10	49.01	54.09	50.00	54.91	49.91	57.99	49.91	14.62	15.53	13.00	15.64	15.96	13.59
v	100	37.07	38.05	31.35	39.36	29.40	42.15	29.40	11.01	11.33	8.47	11.79	12.01	8.63
	200	33.84	40.12	40.48	39.83	41.40	41.40	41.14	10.72	11.29	10.50	11.52	11.30	10.88
	16	29.63	33.37	30.17	34.54	29.93	37.64	29.93	8.70	9.35	8.15	9.33	9.83	8.60
	19	29.89	34.11	29.48	35.22	29.99	38.01	29.99	10.06	9.70	8.30	10.56	10.46	09.07
	20	29.87	33.93	32.46	35.68	32.31	38.41	32.31	8.76	9.54	8.52	11.17	11.07	8.89
	22	31.95	34.02	32.81	34.81	32.01	36.35	32.01	10.15	10.46	8.99	11.29	11.43	8.88
	23	43.46	46.37	38.14	46.15	36.86	45.51	36.86	12.57	13.12	10.46	13.24	12.57	10.14
	31	31.98	35.21	35.81	37.11	36.05	40.35	36.05	10.86	11.59	9.94	11.57	11.27	10.04
	32	34.07	39.25	37.27	40.61	38.30	44.92	38.30	10.99	11.95	10.23	12.05	12.82	11.05
	33	53.29	56.71	50.97	52.96	46.39	53.20	46.39	14.91	14.85	11.29	14.12	13.86	11.29
Average		35.84	39.55	36.32	39.97	42.18	42.18	35.63	10.92	11.35	9.52	11.70	11.71	9.82

configurations of the algorithm, which consists of three methods of initial solutions, two solution representations, and two vehicle loading strategies, were developed. Since no benchmark data sets are not available, a complete set of instances was generated to test these configurations, inspired by existing benchmarks of HFS and VRP from the literature.

Computational evaluations were carried out in two phases. In the first one, the MILP model was executed for small-sized instances, in which 75% of them did not obtain an integer solution after 3600s of executions. The small instances that obtained a result after an hour of execution were compared with the proposed lower bound, obtaining that the lower bound is 32% lower, on average, than the objective function obtained for the best solution

found by the MILP model. In the second phase, an experimental design was performed with 720 generated large instances, and the results were analyzed through the ANOVA statistical test. Seven factors, including four instance factors (number of jobs, stages, customers, and the capacity of the vehicle) and three test factors (initial solution, solution representation, and vehicle loading strategy) were considered in ANOVA as control factors. The response variable was the GAP versus the proposed lower bound. Results showed that all main effects are statistically significant. Due to the assumptions of the ANOVA were not fulfilled, the ANOVA-Type statistic was performed confirming the initial results given by the ANOVA. Particularly, instances with the highest level of stages ($s = 10$) presented the best GAP (14.99%). Also, when the number of customers was set to $c = 31$, the average GAP was 9.92%. When the capacity of vehicles is 100, the GAP presents better performance than when it is set to 200. The computational analysis shows that BR-NEH initial solution, solution representation SR_2 , and loading strategy VLS_1 perform statistically better than the others. It is important to note that, for 48.07% of the instances, the GAP was less than 5%.

Future work could be directed to incorporate various vehicles in the routing to test the best configuration, not only in terms of makespan but also including due date-related measures. Of course, some other solution procedures can be proposed and evaluated.

Chapter 7

Applications in Telecommunication

This chapter ¹ studies the uncapacitated facility location problem (UFLP) in the context of telecommunication systems and smart cities, where a set of facilities must be deployed in order to provide an efficient allocation of processes to them. In this case, the Internet of Vehicles (IoV) scenarios consider the presence of multiple roadside units (RSUs) that should be frequently assigned to operating vehicles. The allocation processes need to be done frequently, quickly, and efficiently, as customer demands change in order to ensure target Quality of Service (QoS) levels. Posteriorly, this problem is extended into a multi-period version of the IoV, where demands are dynamic over time. To cope with both problems, a biased-randomized heuristic is proposed and lately incorporated into an agile optimization framework for dynamically reacting to system changes, which is tested against a set of benchmark instances in order to illustrate its potential.

7.1 The Real-Time Facility Location Problem in Internet of Vehicle Scenarios

The uncapacitated facility location problem (UFLP), initially stated in Balinski (1966), is a popular *NP-hard* optimization problem that, for its very nature, had applications that did not require any strict time constraints in the optimization process, and their inputs were fixed in time. However, over the years, many new domains of application of the UFLP have emerged, some of which require re-optimizing the solution quickly as inputs change slightly but frequently over time. In this context, apart from cost, time becomes an additional factor that affects the decisions in dynamic facility location systems (Boonmee et al., 2017). Hence, the need for smart solving methodologies that can quickly provide sub-optimal solutions has been the subject of research. Examples of such dynamic environments are found in healthcare and relief operations, where demand points, operating and distribution costs can vary over time, and new facilities can be added at different periods (Afshar and Haghani, 2012; Khayal et al., 2015; Moeini et al., 2015). Among different application areas where

¹The contents of this chapter are based on the following works:

- Martins, L. C.; Tarchi, D.; Juan, A.; Fusco, A. (2021): [Agile Optimization for Real-Time Facility Location Problem in Internet of Vehicle Scenarios](#). *Networks*.
- Cesarano, L.; Croce, A.; Martins, L. C.; Tarchi, D.; Juan, A. A. (u.r.): An Agile Optimization Approach for the Multi-Period Internet of Vehicles Problem. *IEEE Access*.

time constraints are crucial, telecommunication systems are one of the most challenging in recent years. The recently introduced 5G communication standard has defined several use cases and applications where low latency communications need to be addressed (Lema et al., 2017). Examples of such applications are the Internet of Vehicles (IoV) (Kaiwartya et al., 2016), Network Functions Virtualization (NFV) (Herrera and Botero, 2016), and the Controller Placement Problem (Lu et al., 2019), in which decisions of *how many* facilities are needed and *where* to place them are taken into account in order to guarantee some users' Quality of Service (QoS).

According to Gerla et al. (2014), the IoV can be defined as a distributed transport network that is capable of making its own decisions about driving customers to their destinations by having communications, storage, intelligence, and learning capabilities to anticipate the customers' intentions. In order to properly work, IoV should be based on a telecommunication infrastructure able to connect every vehicle and node belonging to the IoV system. The vehicular communication infrastructure has been designed in the past years through several communication standards, e.g., dedicated short-range communications (Kenney, 2011) and vehicle-to-everything (V2X) (Zhou et al., 2020). In particular, the V2X standard, developed within the 3GPP standardization body, gained lots of attention since it is based on the presence of four main communication types, i.e., vehicle-to-vehicle (V2V), vehicle-to-infrastructure, vehicle-to-network, and vehicle-to-pedestrian (Technical Specification Group Services and System Aspects, 2019; Technical Specification Group Services and System Aspects, 2020), each one referring to a particular type of communication with different requirements and target scenarios. Within these domains, moving cars need to exchange data with other vehicles, pedestrian or remote nodes directly or through roadside units (RSUs) which are present at fixed locations on the street (Kaiwartya et al., 2016). RSUs can be seen as the interconnecting nodes allowing one to interconnect any element in an IoV scenario. In order to properly design an IoV scenario, several RSUs should be deployed to cover any vehicle while respecting the latency requirements of the desired service (Ji et al., 2020).

Consequently, how to select the number and locations of RSUs to deploy and allocate the traffic load to them is a critical and practical open problem (Ni et al., 2018). In this context, a set of potential RSUs must be strategically selected and activated –i.e., deployed– in order to provide an effective data exchanging network among these moving cars and deployed facilities. Allocating an RSU to a vehicle needs to be done frequently and quickly, as vehicles move through the topology of the streets, and efficiently, as some latency and throughput constraints are required to ensure the QoS (Ni et al., 2018). Moreover, by effectively selecting the RSUs to deploy, the energy consumption of the overall system can be reduced (Wei et al., 2019). Accordingly, this problem is referred to as the Internet of Vehicles Problem (IoVP)

Figure 7.1 represents how the IoVP can be modeled as a UFLP, in which many vehicles in circulation are connected to deployed (in-service) RSUs. This connection makes possible the proper communication among facilities and vehicles in order to keep a target QoS of the system.

The IoV can be formulated as a single or multi-period problem. By considering multiple periods into the problem formulation, this problem becomes of high dynamism since

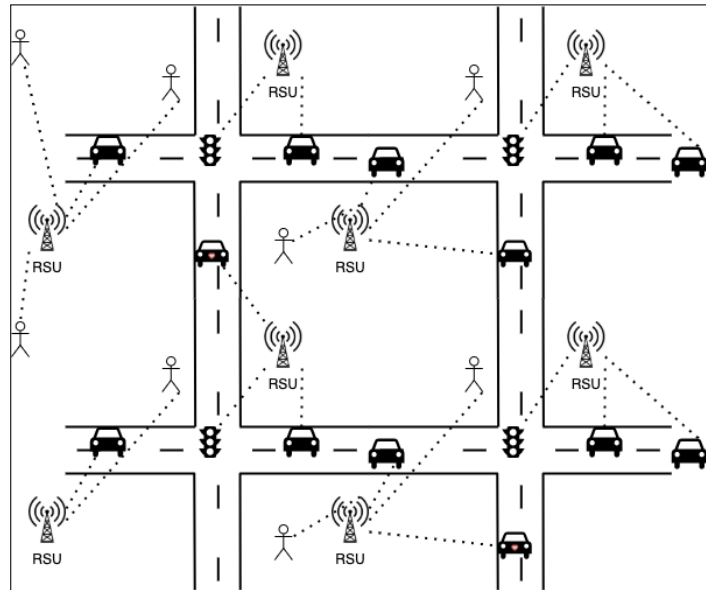


Figure 7.1: The Internet of Vehicles Problem: connection of vehicles in motion to deployed RSUs in a city.

vehicles are constantly moving around the cities. Besides, roads are very dynamic environments: cities may be congested in some areas at specific times of the day, with vehicles moving at a different speed. This dynamism of the environment must be taken into account as well, leading us into a multi-period IoVP (MPIoVP). The MPIoVP can be seen as a rich variant of the IoVP, in which the RSUs configuration has to react and quickly adapt to the ever-changing traffic and connection requirements from the vehicles while keeping the energy consumption low. Thus, every few minutes, a near-optimal configuration of RSUs might need to be re-computed in a short amount of time (typically within a few seconds). In Figure 7.2, the same city blocks are depicted in three different successive periods, where vehicles and pedestrians are in movement and share the same space. Despite the facilities are fixed over time, the users' and vehicles' connections are updated over time –according to the dynamism of this environment– in order to minimize connection costs.

Since the resulting allocation configuration in this type of problem represents a critical impact on the network's performance (Cohen et al., 2015), the need is clear for developing smart and efficient techniques for solving them. In this regard, we extended the savings-based heuristic algorithm, initially proposed by De Armas et al. (2017), into a biased-randomized (BR) algorithm (Grasas et al., 2017), to solve both the IoVP and MPIoVP. By transforming it into a probabilistic algorithm, the exploration of the solution space is enhanced, and, consequently, better solutions can be found. Later, this BR heuristic is incorporated into an agile optimization framework in order to react and adapt to fast-changing customer demands in the system. The AO refers to the massive parallelization of BR heuristics, which are fast in execution and are able to provide high-quality solutions for a range of optimization problems.

The UFLP has been traditionally applied to logistics and supply chains, where decisions are difficult to reverse. On the other hand, in dynamic environments such as virtual resource

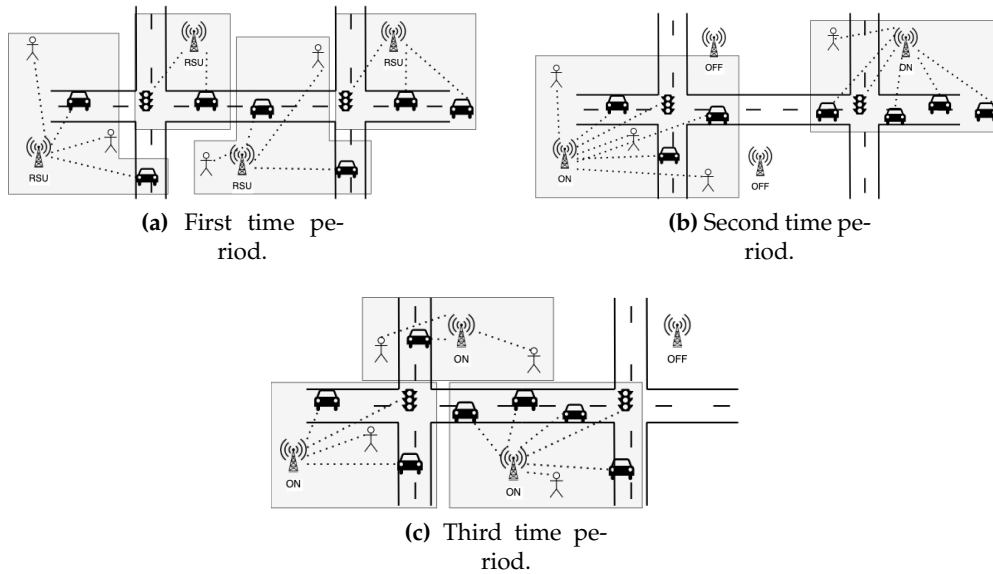


Figure 7.2: The MPIoVP modeled as an UFLP, where user and vehicles share the use of RSUs.

allocation, the costs associated with the opening of a facility are much lower. This allows us to re-optimize the solutions as customer demands change over time by applying the AO framework. In the context of the IoV scenario, it becomes of paramount importance the selection of the best RSU for each vehicle. To this aim, UFLP allows to effectively map the architecture as composed by facilities (i.e., RSU) that should be properly selected by the vehicles while respecting a cost function mapping the communication QoS parameters among the nodes. A series of computational experiments allow us to validate the efficiency of the proposed methodology.

7.1.1 Literature Review

To the best of our knowledge, Balinski (1966) is the first researcher who addressed and explicitly formulated the facility location problem (FLP) as a MIP formulation, initially named as a simple plant location problem (SPLP). At that time, exact approaches for solving FLPs were the most common techniques for many reasons: real-world issues and variants, such as dynamism and stochasticity, were not incorporated into these problems; the problem instances were small- or medium-sized. In this regard, the use of branch-and-bound algorithms for solving the UFLP was largely explored. In Efrøymson and Ray (1966), a branch-and-bound algorithm was proposed for an SPLP, which was later improved by Khumawala (1972), who considered the structure of the FLP for designing an efficient branch-and-bound algorithm. Dual-based branch-and-bound algorithms for solving the UFLP were proposed by Bilde and Krarup (1977) and Erlenkotter (1978). Years later, a multi-level UFLP, where commodities are shipped from origin-level facilities to destination points via a set of intermediate-level facilities, was modeled as a MILP (Tcha and Lee, 1984). Two procedures were applied for accelerating the convergence rate to optimal solutions: node simplification and primal descent procedures. Recently, optimal solutions for the UFLP were published

in De Armas et al. (2017), which were solved through a MIP model using the *Gurobi* solver (Gurobi Optimization, LLC, 2020).

Contrary to exact approaches, approximate methods do not guarantee optimally but are able to generate high-quality solutions in a reasonable amount of time, even for *NP-Hard* problems. In this regard, several heuristics and metaheuristics have been developed for solving the UFLP. Neighborhood search-based methods were introduced in Ghosh (2003) to solve the UFLP, and later, incorporated into a TS metaheuristic and complete local search with memory (CLSM) algorithms in order to escape from local optima. Both TS and CLSM methods returned costs within 0.1% of the optimal. Another TS approach was proposed in Sun (2006), which is guided by a measure of the attractiveness of moves (the opening or closing of a facility). This attractiveness is measured by net cost changes resulting from candidate moves, which are updated (rather than re-computed) from their old values after a move is performed. By updating them, the computation time for solving the problem is largely reduced. At the time, the proposed approach was able to generate better results in less computational time than other methodologies. In Resende and Werneck (2006), the authors developed a two-phase MS metaheuristic to solve the problem. The first stage aims to create randomized solutions by a constructive heuristic, followed by the application of a local search. The second step combines elite solutions by a path-relinking strategy in order to generate high-quality solutions. The proposed method was able to find near-optimal solutions for the majority of existing and generated instances. In De Armas et al. (2017), a fast savings-based heuristic for solving the UFLP was proposed, which is then embedded into an ILS metaheuristic framework. The initial solution for the ILS is generated through a heuristic based on closing opened facilities and re-assigning clients. Also, in Resende and Werneck (2006), the solutions are perturbed by opening facilities and reconstructing the solutions through a path-relinking strategy in the ILS, followed by a local search procedure. Finally, an acceptance criterion is employed to allow accepting worse solutions within a margin limit. Apart from solving the UFLP, the authors also proposed a simheuristic algorithm for solving the stochastic UFLP. This approach combines the proposed ILS with MCS to deal with uncertainty. Near-optimal solutions were found by the ILS in a short computational time. A set of deterministic instances was extended to test the simheuristic approach, which was able to provide different efficient solutions with different trade-offs. In Alvarez Fernandez et al. (2021), the authors model a video streaming with QoS threshold system as a stochastic single-allocation p-hub median problem, by considering a simheuristic solution. A hub-and-spoke network is considered in which a large number of nodes are exchanging real-time multimedia data, and where the quantity of data sent from one node to another is a random variable. Recently, in Hakli and Ortacay (2019), an improved scatter search algorithm to solve the UFLP was proposed. This solution method is called *improved* since different crossover techniques were applied to generate new solutions, and mutation operations improved the local search in the best solution. Consequently, it was able to overcome different population-based solution methodologies, such as those based on swarm optimization and evolutionary algorithms. Among the solution methodologies developed for solving FLPs, we highlight not only the use of exact and approximate (i.e., heuristics and

metaheuristics) methodologies but also the use of approximation algorithms, e.g., Charikar and Guha (1999), Chudak and Shmoys (2003), Jain et al. (2002), and Jain and Vazirani (2001).

As briefly discussed, different problems can be modeled as FLPs. However, it is noticeable the lack of literature on solving the assignment of vehicles to RSUs as a combinatorial optimization problem in the IoV context. As mentioned, in this network, RSUs alongside roads are used as wireless access points, which provide communication coverage to the vehicles inside its coverage area (Kaiwartya et al., 2016). As introduced in Kaiwartya et al. (2016), RSUs alongside roads are used as wireless access points in an IoV network, providing communication services to the vehicles inside its coverage area. Relating this problem to FLPs, the vehicles' demands for communication represent the inputs of our FLP. Since vehicles are in movement, these inputs are, henceforth, dynamic and random, and the outputs need to be periodically recomputed. Dealing with input uncertainties means dealing with real-world scenarios, which leads to the search for more robust or reliable solutions. In Snyder (2006) the authors deal with uncertainties using random variables in its model: cost, demands, and distances. Therefore, RSUs must be deployed in order to establish communication between both parts, i.e., RSUs and vehicles. In Bozorgchenani et al. (2018a), the authors proposed a V2V aided approach for optimizing the task offloading selection in an urban scenario where multiple RSU are supposed to be scattered along the roads, while in Bozorgchenani et al. (2018b), the same authors extended to approach to consider mobile nodes acting as relay and/or processing nodes for implementing on-demand vehicular services. Ni et al. (2018) is one of the few works which proposed an optimization model to solve the RSU deployment problem. They introduced a linear programming-based clustering algorithm which employs a utility function for measuring the total benefit from the RSU deployment. The method is divided into three steps: (i) RSU clustering; (ii) reduction to the single-node instance; and (iii) assigning the tasks, in which (ii) is formulated as the single-node capacitated facility location problem, where there is only one road segment needed to be served by multiple RSUs. Despite addressing a different problem, Khezrian et al. (2014) discussed valuable information regarding the importance of establishing effective communication between RSUs and vehicles. The authors addressed a low-complexity RSU and time slot assignment in vehicular networks with multiple RSUs in tandem, in order to minimize energy consumption and jointly energy-balancing the loading from a normalized energy viewpoint across the RSUs. One reason which supports the first-mentioned finding is that, due to the limited coverage range associated with the RSUs, the average power consumption of an energy-efficient RSU design may be strongly dominated by downlink transmission power. Therefore, RSUs usually prefer to communicate with nearby vehicles instead of those that are more distant. Additionally, they concluded that considering a subset of RSUs to communicate with vehicles, gives additional flexibility in deploying data services whose functionality need not be fully replicated at all RSUs.

In this chapter, a novel approach that models the RSU deployment problem as an UFLP is proposed. Several factors are considered, including the RSU's energy consumption, the service capabilities, and the required quality of service. This model is then solved using an AO method that employs a parallelized version of a biased-randomized algorithm. This

allows us to generate near-optimal solutions in a fraction of time with respect to other optimization approaches.

7.1.2 Problem Definition

As mentioned, several problems from different domains can be modeled as the well-known FLP. In some of these problems, no capacity limit is imposed when assigning customers to the facilities, leading to the UFLP (Cornuéjols et al., 1983). The UFLP is defined by an undirected graph $G = (F, C, E)$, in which F represents the set of facilities, C is the set of customers that must be served from any facility $i \in F$, and E is the set of edges which connect facilities with customers. Each facility $i \in F$ is characterized by an opening cost f_i , and the cost associated with the assignment of customer j to facility i is given by $c_{ij} \geq 0$, which is usually represented by the distance from client i to the facility j , where the requested services are provided. The facilities are uncapacitated in the sense that no customers' assignment limit constraint is imposed for each facility $i \in F$. Notice that this model assumes that the RSUs have an unlimited capacity. This might not be a fully realistic assumption due to the generation of delays and the loss of QoS, however, this relaxation allows us to validate the proposed algorithm against the results in the literature. Let x be the decision variable that indicates when a customer j is assigned to facility i . Therefore, $x_{ij} = 1$ if customer j is served by facility i , and $x_{ij} = 0$, otherwise. Accordingly, the binary variable y is used to represent when a facility i is opened ($y_i = 1$), incurring its respective opening cost f_i . Given this formulation, the UFLP can be modeled as:

$$\min \sum_{i \in F} \sum_{j \in C} c_{ij} x_{ij} + \sum_{i \in F} f_i y_i \quad (7.1)$$

subject to:

$$\sum_{i \in F} x_{ij} = 1 \quad \forall j \in C \quad (7.2)$$

$$x_{ij} \leq y_i \quad \forall i \in F, \forall j \in C \quad (7.3)$$

$$y_i \in \{0, 1\} \quad \forall i \in F \quad (7.4)$$

$$x_{ij} \in \{0, 1\} \quad \forall i \in F, \forall j \in C \quad (7.5)$$

In this regard, the objective when solving the UFLP is to minimize the fixed and variable costs of the network G , given by: (i) the opening cost of the serving facilities; plus (ii) the respective serving cost from them to their customers. Constraints (7.2) guarantee that every client must be satisfied by a single facility, which must be open (7.3). Finally, Constraints (7.4) and (7.5) state that all decision variables are binary.

7.1.2.1 The Single-Period Internet of Vehicles Problem

Despite being naturally of large applicability in many contexts, the classical UFLP has been frequently adapted to properly address its different definitions according to the application

contexts. This adaptation refers, mainly, to alternative ways of calculating the costs for opening facilities and for assigning clients to them according to the environment specifications.

An IoV scenario is composed of the presence of several RSUs placed in an urban environment and acting as connection points for the vehicles moving in the area. RSUs are exploited by the vehicles for implementing any of the previously mentioned V2X services. When selecting the best RSUs among those available in the area, a vehicle should take into account its own QoS requirements, which should be respected through a proper RSU selection. The goal of our approach is to jointly consider QoS communication characteristics, in terms of delay and throughput, and energy consumption, for developing a UFLP based strategy aiming at jointly mapping the RSUs to vehicle selection while keeping the number of active RSUs as reduced as possible in order to reduce the energy consumption.

With this in mind, the UFLP formalization is considered in the context of the IoV, by introducing proper values for the assignment costs c_{ij} and the opening costs f_i , modeled in terms of QoS and energy consumption, respectively.

7.1.2.1.1 Opening Cost Model In the UFLP, the opening cost of a facility (i.e., a RSU) corresponds to the cost incurred in using that specific facility. When considering an IoV scenario, the opening cost corresponds to the cost incurred in having that specific RSU as working, i.e., its energy consumption. In the following, we consider that an RSU is not consuming energy when no vehicle is allocated to it, corresponding to assuming that the RSU could be turned off, while energy consumption occurs if any of the vehicles are connected to the RSU while they are ranging between a sleep and an active state (Han et al., 2016). Therefore, we can consider the energy spent in a sleep state as a floor, hence, the opening cost represents the difference between the energy spent in the active and in the sleep states. In our model, the opening cost can be defined through an ON/OFF energy factor E_i of the RSUs. Since no composition is required, we can simply define:

$$f_i = E_i \quad (7.6)$$

7.1.2.1.2 Assignment Cost Model The assignment cost c_{ij} can be modeled as the quality of the connection between a vehicle j and an RSU i . In a real-world environment, this value is not only tied to the actual throughput S_{ij} and delay t_{ij} of the $j \rightarrow i$ connection, but also to the target throughput \bar{S}_j and the target delay \bar{t}_j of the specific type of service requested by the vehicle j . It should be noted that the values S_{ij} and t_{ij} are dependent on: (i) the distance between the vehicle and the RSU; (ii) the propagation environment of each RSU-to-vehicle link, and (iii) the RSU's hardware capabilities. Furthermore, the modeling of c_{ij} should also take into consideration a minimum guaranteed level of service, specified by the variables S_j^{min} and t_j^{max} . In case these constraints are not respected, the connection between i and j should have the highest cost, and the corresponding c_{ij} should be set to 1.

According to these criteria, we define two cost functions that will be later composed to obtain c_{ij} . The purpose of these cost functions is to assign, to each parameter (i.e., the actual throughput and delay), a value in the $[0, 1]$ range that represents the fitness of the

parameter with respect to the user's needs. In order to take into account a target service value, we resort to a logistic function, whose application is often considered for modeling QoS user satisfaction in wireless communications (Mazza et al., 2014). To this aim, two logistic functions have been defined for mapping the throughput and the delay:

1. Throughput cost function:

$$g_1(S_{ij}) = \frac{1}{1 + e^{-\alpha_1(S_{ij} - \bar{S}_j)}}$$

2. Delay cost function:

$$g_2(t_{ij}) = \frac{1}{1 + e^{-\alpha_2(t_{ij} - \bar{t}_j)}}$$

where α_1 and α_2 drive the steepness of the curves, and will be set in order to have $g_1(S_j^{min}) \approx 1$ and $g_2(t_j^{max}) \approx 1$, i.e., the cost is maximized when the throughput is lower than the minimum, and the delay is higher than the maximum. In the model, \bar{S}_j and \bar{t}_j represent the points where the marginal gain is maximized. The overall quality of service in the $[0, 1]$ range is then obtained by composing the two cost functions through a weighted sum. The parameters w_1 and w_2 are designer specific variables for giving more importance to either of the QoS variables (Q):

$$Q_{ij} = w_1 \cdot g_1(S_{ij}) + w_2 \cdot g_2(t_{ij})$$

The assignment cost c_{ij} should also consider as an additional parameter the energy cost related to the link establishment. By resorting to the Small Base Station (SBS) power consumption model presented in Han et al. (2016), where the consumed power in active mode is a function of two terms where one is proportional to the load of the SBS, we introduce E_i^{oh} , which is the energy overhead of the i -th RSU, associated with the establishment of a connection, including both link setup and transmission. Since QoS and Energy are semantically different, we introduce a parameter γ that allows us to define the overall trade-off between operating cost and service reliability; γ should be tuned experimentally according to the wanted behavior of the algorithm, leading to the link cost formulation:

$$c_{ij} = \gamma \cdot Q_{ij} + E_i^{oh}$$

7.1.2.1.3 Objective Function We are now in possession of every building block to rewrite the objective function (7.1) of the UFLP as:

$$\min \sum_{i \in F} \sum_{j \in C} \left(\gamma Q_{ij} + E_i^{oh} \right) x_{ij} + \sum_{i \in F} E_i y_i \quad (7.7)$$

7.1.2.2 The Multi-Period Internet of Vehicles Problem

The single-period IoVP is considered static in the sense that no dynamism is incorporated into the system. However, the IoVP cannot be modeled with a static assignment of RSUs and

vehicles due to vehicular mobility. Therefore, the MPIoVP is a version of the well-known IoV problem (Yang et al., 2014), in which a dynamic evolution of the traffic is considered over time. In particular, we are interested in taking into account changes in the vehicles' location, which might affect the aggregated demand associated with each cell.

For taking into account the dynamicity of the system, an entire area \mathcal{A} is divided into J geographical sub-areas, named *cells*, where the j th cell is identified as \mathcal{C}_j , and $\mathcal{A} = \cup_j \mathcal{C}_j$. Within each cell, multiple vehicles are located at a given time t , where:

$$\mathcal{V}_j^{\mathcal{C}}(t) = \{v_m | v_m \in \mathcal{C}_j\} \quad \forall v_m \in \mathcal{V}$$

is the set of vehicles within the j th cell, at time t . In order to map the overall requests of the vehicles within a cell, we introduce a demand factor d_j^t corresponding to the demand of the vehicles within the j th cell at time t . The demand can be considered as an overall factor mapping the requested data rate, amount of services, and connection quality. The higher is the demand, the more resources should be reserved for the vehicle from a certain RSU. We consider having a random demand in each cell, whose value is changing in time. Henceforth, the system must be re-optimized taking into consideration the vehicle mobility.

With this in mind, an assignment cost based on the distance between a cell and an RSU is considered, together with the demand of the cell itself. The total demand in a cell is the sum of all the individual demands of the vehicles currently located in that cell. Since the vehicles are continuously in motion, the demand in each cell varies over time. Considering an initial setting of demands, the goal is to determine which is the lowest-cost configuration of RSUs, i.e., which RSUs need to remain operative in order to minimize the sum of energy and delay costs.

The generic RSU_i at time t can be characterized by a cost $\lambda_i^t r_i$, depending on its status (i.e., operative or not) in the previous period $t - 1$. We define r_i as the maximum amount of energy that can be consumed, and λ_i^t as a weight factor depending on the state in the previous period. If the i th RSU was off at $t - 1$, the cost at t is considered equal to $\lambda_i^t r_i$, with $\lambda_i^t = 1$. Otherwise, only a partial cost $\lambda_i^t r_i$ (i.e., a maintenance cost) is paid, where $0 < \lambda_i < 1$. Thanks to this, we are able to map a higher cost for switching on the RSU rather than maintaining it operative. By setting a decision variable y_i^t , corresponding to the selection of a certain RSU_i at time t , as:

$$y_i^t = \begin{cases} 1 & \text{RSU}_i \text{ is active at time } t, \\ 0 & \text{otherwise.} \end{cases} \quad \forall RSU_i \in \mathcal{R}, \forall t \quad (7.8)$$

it is possible to define the overall cost related to the activation of the RSUs in any time period, whose value should be minimized, as:

$$\min \sum_{RSU_i \in \mathcal{R}} \lambda_i^t r_i y_i^t \quad \forall t \in T \quad (7.9)$$

While the cost in (7.9) is related to a fixed cost to be paid for either keeping operative or activating the RSUs, in order to take into account the load to be managed by each RSU –in

terms of vehicles–, an additional cost term should be introduced for taking into account the distance between the RSUs and the cells, modeling the link availability and quality between RSUs and vehicles. Notice that this value is fixed over time, while, as later explained the vehicles and their related demand is going to change within each cell.

Let us consider a cost matrix \mathbf{C} , where a generic element c_{ij} stands for the cost between RSU_i and a cell j in terms of distance, so that $c_{ij} \propto d_{ij}$ (i.e., the higher the distance, the higher the cost), where d_{ij} is the distance between the i th RSU and the j th cell.

Since each cell represents the geographical sub-area in which the vehicles are positioned, we have to map the vehicle density and their communication needs for each cell. In order to do this, we introduce a demand vector, identified as \mathbf{D} , which indicates the aggregated demand per cell. To be more specific, in order to consider the dynamic behavior of the system, we identify with $\mathbf{D}^{(t)}$ the demand vector at time t whose elements d_j^t represents the demand of a cell j at time t . The higher the number of vehicles in that cell, the higher the associated demand. The total demand of a cell j in a period t is the sum of the individual demands of the vehicles in that cell, i.e.:

$$d_j^t = \sum_{v_m \in \mathcal{C}_j} \tilde{d}_m^t \quad \forall \mathcal{C}_j \in \mathcal{A}, \forall t \in T$$

where \tilde{d}_m^t is the individual demand of the m th vehicle at time t .

Let us introduce now a decision variable x_{ij} standing for the connection between the i th RSU and the j th cell in time period t as follows:

$$x_{ij}^t = \begin{cases} 1 & \text{RSU}_i \text{ is connected to cell } j \text{ in period } t, \\ 0 & \text{otherwise.} \end{cases} \quad \forall RSU_i \in \mathcal{R}, \forall \mathcal{C}_j \in \mathcal{A}, \forall t \in T \quad (7.10)$$

corresponding to the possibility to connect all the vehicles in the set $\mathcal{V}_j^{\mathcal{C}}(t)$, i.e., belonging to the j th cell at time t , with the i th RSU. Therefore, for a given period $t \in T$, we can model the MPIoVP as:

$$\min \left\{ \sum_{RSU_i \in \mathcal{R}} \lambda_i^t r_i y_i^t + \sum_{RSU_i \in \mathcal{R}} \sum_{\mathcal{C}_j \in \mathcal{A}} c_{ij} d_j^t x_{ij}^t \right\} \quad (7.11)$$

subject to:

$$\sum_{\mathcal{C}_j \in \mathcal{A}} x_{ij}^t = 1 \quad \forall RSU_i \in \mathcal{R}, \forall t \quad (7.12)$$

$$x_{ij}^t \leq y_i^t \quad \forall RSU_i \in \mathcal{R} \quad (7.13)$$

$$x_{ij}^t, y_i^t \in \{0, 1\} \quad \forall RSU_i \in \mathcal{R}, \forall \mathcal{C}_j \in \mathcal{A} \quad (7.14)$$

$$r_i, c_{ij}, d_j^t \in \mathbb{N} \quad \forall RSU_i \in \mathcal{R}, \forall \mathcal{C}_j \in \mathcal{A} \quad (7.15)$$

where:

- The objective function in (7.11) should be minimized for having both RSU activation and cell allocation to each RSU minimized;
- Constraints (7.12) ensure that, inside each period, each cell j can be associated with only one RSU i ;
- Constraints (7.13) guarantee that, if there is a link between a cell j and an RSU i , then the RSU is operative;
- Constraints (7.14) indicate that the two decision variables are binary;
- Constraints (7.15) stands that r_i, c_{ij}, d_j^t are integer variables.

7.1.2.2.1 Demands Formulation The higher the number of vehicles in a cell is, the higher the weight of a connection between that cell and the closest RSU. We model such condition through the demand \mathbf{D} , representing the number of connections required by all vehicles in that cell. However, since the urban environment is dynamic, the demand varies over time, which introduces the necessity of a fast heuristic algorithm that can easily adapt the dynamic RSUs' configuration. The second member of the objective function in (7.11) takes into account the demands vector \mathbf{D} associated with the cells of the system. At the beginning of every time interval, the demand is supposed to change with a probabilistic pattern, modeling the vehicle behavior within an urban area. More details on how this occurs are provided next.

7.1.2.2.2 The Shifting Model For modeling the movement of vehicles inside a city, a graph representation of the area, $G = \{E, V\}$, is considered, where the vertexes represent the cells, and the edges represent the demand shift among adjacent cells. In order to have a realistic model, the following hypotheses when setting up the model were taken into account:

- The demand vector has the same cardinality as the number of cells in the topology;
- Demands can shift only among adjacent cells, including the perimeter of the region being considered;
- The total demand is constant over time, meaning that when part of the demands leave the system (i.e., crosses the perimeter), an identical quantity of demand enters from the opposite border, i.e.:

$$\sum_{C_j \in \mathcal{A}} d_j^t = cost \quad \forall t \in T$$

This model can effectively consider a realistic urban area scenario. Although our methodology can be applied in any other scenario, in our computational experiments, we will assume the existence of a high-density area, namely *downtown area*, which attracts vehicles

(i.e., demands) in certain periods of the day (e.g., business hours) and from which vehicles departure in other periods (e.g., once most business close).

Pseudocode 20 presents the shifting model, followed by the shifting function in Pseudocode 21, and the probability rule that every cell is subjected to. If a shifting probability is met, each cell would transfer a shifting quantity value towards the downtown area when:

$$Pr(move) \leq \alpha(to\ Downtown) + (1 - \alpha)(from\ Downtown)$$

If the previous condition is not met, the cell transfers its shifting quantity value towards another random direction, excluding the downtown area direction.

Pseudocode 20: Pseudo algorithm of the shifting model (main flow)

Data: destination area *downtown*, list of demands *Demands*, $randAlpha \in (0,1)$, $randShift \in (0,1)$, probability that a cell shifts $shiftingProbability \in (0,1)$, quantity of demand that shifts across cells $shiftingQuantity \in [0,1]$

```

1 Function shiftingProcedure(downtown, Demands, randAlpha, randShift,
  shiftingProbability, shiftingQuantity):
2   foreach d in Demands do
3     if randShift  $\leq$  shiftingProbability then
4       if randAlpha  $\leq$  alpha then
5         | shiftToward(Demands, d, downtown, shiftingQuantity)
6       else
7         | stepFurther(Demands, d, downtown, shiftingQuantity)
8       end
9     end
10  end
11  return
12 End

```

Pseudocode 21: Pseudo implementation of the *shiftToward* function.

Data: destination area *downtown*, list of demands *Demands*, demand *d*, quantity of demand that shifts across cells $shiftingQuantity \in [0,1]$

```

1 Function shiftToward(downtown, Demands, d, shiftingQuantity):
2   if d > downtown then
3     | Demands[d - 1]+ = ceil(shiftingQuantity * Demands[d])
4     | Demands[d]- = ceil(shiftingQuantity * Demands[d])
5   else
6     | Demands[d + 1]+ = ceil(shiftingQuantity * Demands[d])
7     | Demands[d]- = ceil(shiftingQuantity * Demands[d])
8   end
9   return
10 End

```

In a real scenario, demands may be refreshed, in each time interval, by: (i) exploiting an external database to be periodically queried; (ii) using RSU sensing capabilities to estimate the demands in real-time; and (iii) a prediction model that forecasts the new values.

7.1.2.2.3 A Numerical Example In order to verify the effectiveness of the demand shifting model, we have carried out a numerical experiment using the parameters presented in

Table 7.1, where the cells are organized as a 4×4 square.

Table 7.1: Parameters setting for the numerical example of demands shifting.

Parameters	Values
<i>number of periods</i>	20
α	0.75
<i>downtown area (in bold)</i>	[2, 2]
<i>shifting probability</i>	50%
<i>shifting quantity</i>	40%
<i>cells</i>	16

Let us assume that the initial demands in period $t = 0$ are given by the following matrix, which is a more effective geographical representation of the area. The bold element represents the downtown area:

$$\mathbf{D}^{(0)} = \begin{bmatrix} 5 & 5 & 5 & 5 \\ 5 & 5 & 5 & 5 \\ 5 & 5 & \mathbf{5} & 5 \\ 5 & 5 & 5 & 5 \end{bmatrix}$$

Based on this initial configuration and the aforementioned parameters setting (Table 7.1), a potential shifting during the next period is:

$$\mathbf{D}^{(1)} = \begin{bmatrix} 3 & 3 & 5 & 4 \\ 3 & 3 & 5 & 8 \\ 10 & 7 & \mathbf{5} & 5 \\ 5 & 4 & 5 & 5 \end{bmatrix}$$

Similarly, after successive shifting processes, the configuration at $t = 19$ is:

$$\mathbf{D}^{(19)} = \begin{bmatrix} 0 & 2 & 2 & 0 \\ 0 & 0 & 4 & 1 \\ 1 & 31 & \mathbf{33} & 0 \\ 0 & 2 & 4 & 0 \end{bmatrix}$$

To better understand the concept shown in a matrix form, Figure 7.3 displays, accordingly, a geographical heatmap of a neighborhood belonging to the city of Rome, Italy. The heatmap in Figure 7.3a represents the initial scenario, where all the demands are well distributed all over the considered area. As time evolves, the demands shift towards the downtown area.

7.1.3 Solution Method: From a Heuristic to a Agile Optimization Framework

Since the UFLP is *NP-Hard*, the use of exact methods is not an efficient strategy to solve large-scale instances in short computing times. Moreover, our scenario requires a real-time

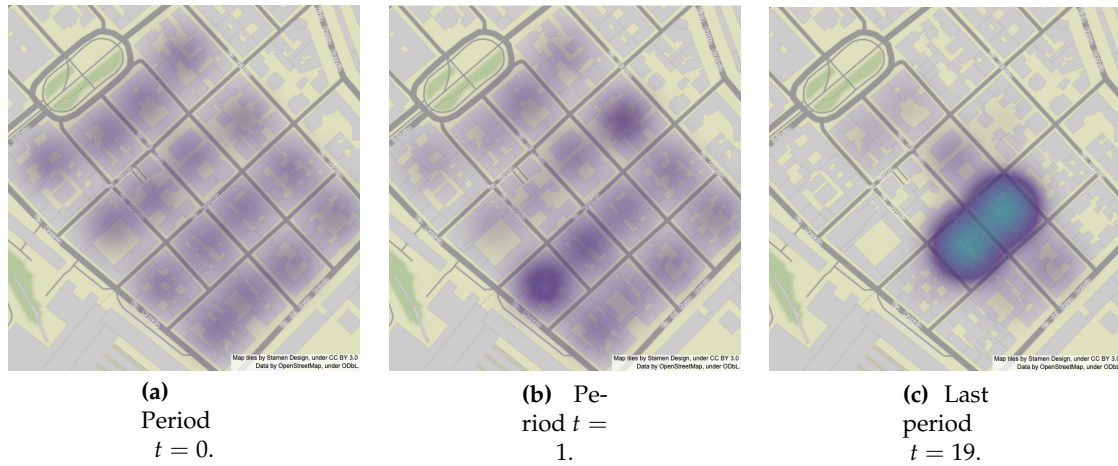


Figure 7.3: Heatmaps of traffic density at different time periods.

approach for practical implementation. Therefore, to solve both the single- and multi-period IoVP, we employ an extended version of a savings-based heuristic to solve the UFLP. This heuristic is later enriched with biased-randomization techniques and embedded into an agile optimization framework. The use of this framework allows the re-optimization of the system every time a piece of new information should be incorporated into the model, which represents a typical environment when dealing with these problems in the field of telecommunication. As we will discuss, the proposed heuristic is able to generate good-quality solutions in just a few seconds or even less.

7.1.3.1 Approaches for the Single-Period Internet of Vehicles Problem

7.1.3.1.1 The Savings-Based Heuristic Our proposed approach extends the savings-based heuristic proposed in De Armas et al. (2017) to solve the UFLP. This heuristic is based on the idea of closing opened facilities according to an associated saving cost and then re-assigning the previously allocated customers to the remaining opened ones. The savings cost of closing an open facility is given by: (i) the cost of opening it; plus (ii) the assignment cost of its customers; minus (iii) their reallocation cost to alternative facilities. Initially, all the facilities are open, and the initial savings costs are calculated for each one (line 1). The list of potential closings is sorted in descending order (line 2), and, originally, the top element—the one with the highest saving cost—with positive saving value is selected, and the respective facility is closed (line 8). Since a negative saving cost implies a more expensive allocation setting than the previous one, it is automatically discarded, hence, only positive savings are accepted (line 7). The later stages regard the steps of determining affected customers (line 9) when closing the chosen facility and their re-assignment to the remaining alternative opened facilities (line 10), as well as the re-computation of the individual savings given the new allocation scenario (line 11). The new saving list is then re-sorted (line 12), and this process is repeated while the savings list is not empty (line 3). Because only a subset of affected customers should be re-allocated to alternative facilities, this process is

computationally cheaper than an opening-facilities-based strategy, where the complete set of customers must be considered in the allocation process whenever a facility is opened.

7.1.3.1.2 Extending to a Biased-Randomized Algorithm As the original heuristic does not incorporate any randomness to guide the search, it always generates the same solution when starting from the same point. However, although being reasonably efficient, this strategy does not provide an efficient and effective exploration of the solution space. In order to change this behavior, this heuristic is extended into a biased-randomized strategy by incorporating a skewed probability distribution in the constructive procedure. As discussed in previous studies (Juan et al., 2013a), this allows us to transform the deterministic heuristic into a probabilistic algorithm without losing the logic behind the original heuristic.

Following previous research work on biased-randomization techniques (Grasas et al., 2017), we have employed the geometric probability distribution to ‘induce’ this biased-randomized behavior during the solution-construction process. This distribution has been chosen since it offers some convenient properties, such as: (i) it only uses one parameter, β , which is easy to set since $\beta \in (0, 1)$; (ii) by varying the value of β , different degrees of randomness can be considered, e.g., values close to 0 emulate a uniform-random behavior, while those close to 1 emulate a greedy one; and (iii) generation of random variates from a geometric probability distribution is extremely fast since there are analytical expressions that allow us to obtain them –thus avoiding the use of time-expensive loops. Other skewed probability distributions can be also employed, e.g., the descendent triangular (Grasas et al., 2017). However, the geometric probability distribution offers a higher degree of flexibility without incurring in complicated fine-tuning processes. As a consequence, it is the one utilized by most of the biased-randomization algorithms published so far.

As discussed in Section 2.3, biased-randomization techniques allow us to generate different good-quality solutions every time the algorithm is executed. Notice that these executions are independent of each other, so they can be run in parallel. By doing it, we can obtain high-quality solutions (improving the one originally provided by the heuristic) with virtually the same wall-clock time as the one employed by the heuristic itself –which might be in the order of milliseconds. This integration of biased-randomization techniques with parallel computing becomes then an agile optimization strategy.

Pseudocode 22 describes the extended heuristic, based on De Armas et al. (2017)’s approach, in which a geometric distribution randomizes (smooths) the selection stage (line 5), which is greedy in the original heuristic.

7.1.3.2 Approaches for the Multi-Period Internet of Vehicles Problem

7.1.3.2.1 The Extended Savings-Based Heuristic Similar to the extended heuristic proposed for solving the IoVP (Pseudocode 22), the current heuristic is based on the concept of switching off facilities according to the savings, which is used to guide the search for feasible solutions. This procedure is shown in Pseudocode 23. For the MPIoVP, new decisions have to be accounted according to the weight factor λ_i whose objective is to introduce a

Pseudocode 22: Extended Heuristic for solving the IoVP

Data: set of facilities F , set of customers C , set of edges E , geometric distribution parameter β , initial solution as a set of open facilities $sol \subseteq F$

```

1 Function BRH( $F, C, E, \beta, sol$ ):
2    $L \leftarrow \text{createSavingsList}(sol)$ 
3    $L \leftarrow \text{sort}(L)$ 
4   while  $L$  is not empty do
5     Randomly select position  $x \in \{1, \dots, |L|\}$  according to distribution  $Geom(\beta)$ 
6      $f \leftarrow \text{selectTheXthFacilityFromList}(L)$ 
7      $savingCost \leftarrow \text{getSavingCostOfFacility}(f)$ 
8     if  $savingCost > 0$  then
9        $sol \leftarrow \text{closeFacility}(f)$ 
10       $C' \leftarrow \text{getAffectedCustomers}(sol)$ 
11       $sol \leftarrow \text{assignCustomersToOpenedFacilities}(C')$ 
12       $L \leftarrow \text{updateSavingsList}(sol)$ 
13       $L \leftarrow \text{sort}(L)$ 
14     end
15      $\text{deleteFacilityFromList}(f, L)$ 
16   end
17 End

```

maintenance cost according to the state of an RSU i in the previous period, and the inclusion of dynamic demands into the system. In other words, the cost of each facility is now updated based on their state in the following time period (i.e., if it is switched on, switched off, or remains in the on state, lines 2-7). The next steps are based on selecting an operative facility, from the savings list, according to its respective *saving* and switching it off. Accordingly, its previously allocated cells must be re-assigned to the closest operative RSUs. In this case, the operation of computing the savings value (line 8) is performed for every operative facility as follows: (i) the cost of switching on / maintaining a facility; plus (ii) the assignment cost of its customers multiplied by their demand; minus (iii) their reallocation cost to alternative facilities multiplied by their demand. Notice that both the maintenance cost and demands are now added in the savings calculation in order to reflect the needs of the new multi-period system. The list of savings is sorted in descending order (line 9), and the facility with the highest positive savings (in case of a greedy behavior, or $\beta = 1$ for the $Geom(\beta)$ probability distribution, for example) is selected (line 12) to be switched off (line 15). Since negative savings values imply costly allocation settings, they are discarded. Once a facility is switched off, the affected cells are re-assigned to alternative RSUs (line 17), which implies a re-computation of the individual savings given the new allocation scenario (line 18). Once re-calculated, the savings list is re-sorted (line 19), and this process is repeated until the end of the sorted list.

The described heuristic is originally applied for a single period. However, by introducing the shifting demands on the MPIoVP, as described in Section 7.1.2.2.1, running the heuristic only for the first period is not efficient and enough, since RSU deployments and vehicles' assignment might change over time. In this way, the following different approaches are considered, depending on how this heuristic is employed to solve the MPIoVP:

Pseudocode 23: Extended Heuristic for solving the MPIoVP

Data: set of facilities F , set of customers C , set of edges E , geometric distribution parameter β , initial solution as a set of open facilities $sol \subseteq F$, the configuration from previous period sol_t

```

1 Function BRH( $F, C, E, \beta, sol$ ):
2   foreach facility  $i$  in  $sol$  do
3     if  $i$  is open in  $sol_t$  then
4       | e.g.  $\lambda = 0.5 * \lambda$ 
5     end
6      $r'_i \leftarrow \lambda * r_i$ 
7   end
8    $L \leftarrow \text{createSavingsList}(sol, d_j^{t+1}, r'_i)$ 
9    $L \leftarrow \text{sort}(L)$ 
10  while  $L$  is not empty do
11    Randomly select position  $x \in \{1, \dots, |L|\}$  according to distribution  $\text{Geom}(\beta)$ 
12     $f \leftarrow \text{selectTheXthFacilityFromList}(L)$ 
13     $\text{savingCost} \leftarrow \text{getSavingCostOfFacility}(f)$ 
14    if  $\text{savingCost} > 0$  then
15      |  $sol \leftarrow \text{closeFacility}(f)$ 
16      |  $C' \leftarrow \text{getAffectedCustomers}(sol)$ 
17      |  $sol \leftarrow \text{assignCustomersToOpenedFacilities}(C')$ 
18      |  $L \leftarrow \text{updateSavingsList}(sol, d_j^{t+1}, r'_i)$ 
19      |  $L \leftarrow \text{sort}(L)$ 
20    end
21     $\text{deleteFacilityFromList}(f, L)$ 
22  end
23 End

```

1. *The single-open static approach:* in this strategy, a single facility, usually located in the downtown area, is kept on.
2. *The all-open static approach:* in this configuration, every facility is operative at any time period. Hence, cells will always be assigned to their closest RSU (which will be always on). However, this strategy is highly costly since all the facilities are operative in every period.
3. *The keep-the-first static approach:* in this scenario, the system will be optimized only once, in the first period. Thereafter, the first configuration of RSUs is kept for the rest of the periods, while the variable cost in each period is updated according to the new demands and the cost of keeping operative the selected facilities. In principle, this strategy will be less costly than the previous one. However, it does not adapt to variations in the vehicles' demands.
4. *The partially-dynamic approach:* as an enhancement of the previous scenario, in this one, we re-optimize the system's configuration after each set of k periods, with $k > 1$. In order to apply this strategy in practice, it becomes necessary to provide a good balance between computational time and the quality of the recommended configuration.

5. *The fully-dynamic approach*: this is a similar scenario as the previous one, but when $k = 1$. For achieving this, an agile strategy is needed in order to find a near-optimal configuration in each period and in real-time. We can expect this approach to provide better results than the previous one, although it requires a considerable effort in terms of quickly re-optimizing the system after periods that might span over just a few minutes each.

Figure 7.4 depicts the high-level flow of the proposed algorithms. The main steps are: (i) initialize the system (only the first period); (ii) recover the demands values; and (iii) apply a constructive heuristic for the generation of a new solution. Steps (ii) and (iii) are repeated in each time period. Due to the MPIoVP dynamism, we are now focused on the dynamic approach that integrates the described savings-based heuristic.

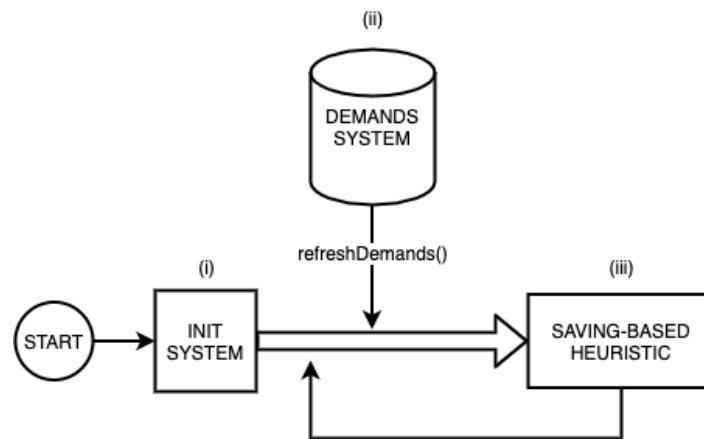


Figure 7.4: General Architecture of our approach.

7.1.3.3 Agile Optimization

As described, some applications, such as the IoV –which includes the single-period IovP and MPIoVP–, require the assignment of customers to facilities –e.g., vehicles to RSUs– in real-time in order to meet customers’ demands. Some of them also require the processing of new information dynamically in real-time, when some resources are already allocated to customers in previous requests (Fan et al., 2015). In this regard, a re-optimization of the system is required whenever a new piece of information should be incorporated into the model, such as allocating new resources to new clients as well as changing previous decisions in order to include, remove, or update new demands. In this way, the need for real-time decision-making is clear.

As described in Section 2.5, the AO takes advantage of combining powerful approaches from both parallel computing and biased-randomization of heuristics. For being extremely fast in execution, easily parallelizable, flexible, and requiring the fine-tuning of only a single parameter (in our case, the β), the combination of BR techniques with parallel computing allows the finding of reasonably high-quality solutions for a range of large-scale and *NP-hard* optimization problems in real-time. In the AO framework, several independent executions

of a BR heuristic –each one constituting a thread– is run concurrently. Consequently, many alternative different solutions are generated in the same wall-clock time as the one employed by the original heuristic. Some of these solutions outperform the one generated by the original heuristic, and the best-found solution is returned. For being able to generate solutions in real-time, both the solution approaches –to solve the IoVP and MPIoV, respectively– are embedded into an AO framework.

7.1.4 Computational Experiments and Results

To test and validate our approaches, we have used the MED benchmark instances, which were originally proposed for representing the p -median problem (Ahn et al., 1988). Years later, some authors employed the same benchmark in the UFLP context (Barahona and Chudak, 2005), being them largely studied thenceforth. These instances are characterized by a different number of locations –being each location a potential RSU to be switched on or a cell, simultaneously– and also by the cost of switching on of each RSU, which follows pre-defined number scales. For each of the six possible problem magnitudes (500, 1000, 1500, 2000, 2500, and 3000 locations), three different switching on costs schemes are considered: 10, 100, and 1000, in which the higher the switching on cost scheme, the cheaper the switching on of an RSU. Accordingly, the switching cost of an RSU i is given by $r_i = \sqrt{n}/sw \times 10000$, where sw represents the switching on costs scheme, and n is the number of cells from the system. For example, the instance 500-10 considers a switching cost of $r_i = \sqrt{500}/10 \times 10000 \approx 22361$ for every RSU i . The computational environment employed for evaluating our approaches is an Intel Core *i7-8550U* processor with 16 GB of RAM.

7.1.4.1 Experimental Design

As aforementioned, the power of our agile optimization approach lies in the heavy parallelization of BR heuristic executions, whose performance is related to its parameters. In our case, a geometric distribution, which is controlled by the parameter $\beta \in [0, 1]$, was used for introducing the random behavior in its solution space exploring process. When $\beta = 1$, the original greedy heuristic behavior is kept, while $\beta = 0$ refers to the completely random process. On the other hand, our agile optimization framework is implemented by employing a fixed number of independent worker threads, each executing the biased-randomized heuristic characterized by a different beta value.

For setting β , different intervals were considered. For each 0.1 interval range, from 0.0 to 1.0, 10 executions were performed for each problem instance. Figures 7.5a and 7.5b show the convergence of the solutions over different β intervals, for problem instances 2500-10 and 3000-10, which are composed of 2500 and 3000 facilities, respectively. They are compared over the solution generated by the deterministic approach (orange line) and the optimal solution obtained through Gurobi solver (green line).

As we can see, when employing intermediate values of β , it's more likely to result in better solutions. Concretely, this behavior is noticed when β is chosen between $\beta \in [0.4, 0.8]$.

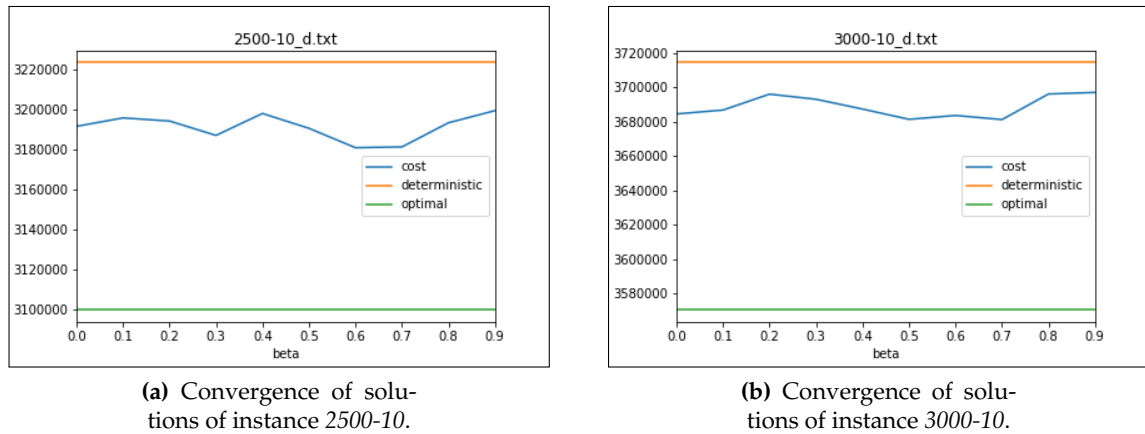


Figure 7.5: Results obtained when varying the β value from 0 to 1.

Since the remaining instances showed similar behavior, therefore, we set this interval to randomly choose the parameter β , due to its capability to generate lower-cost solutions when selected in this range.

Once β is defined, we simulate the parallelization of these threads by sequentially running the biased-randomized executions, each one using a different seed for the pseudo-random number generator, and take the slowest run's execution time as the wall clock time of the overall execution. The next two sections present and discuss the results for both the single- and multi-period IoV problems, respectively.

7.1.4.2 Results for the Single-Period Internet of Vehicles Problem

In Table 7.2, we address the parameters setting for the computational experiments regarding the IoVP formulation (Mazza et al., 2018; Mazza et al., 2014; Yu et al., 2014), as introduced in Section 7.1.2.1. It is worth to be noticed that the opening cost values for E_i for the different instances, derived from the amount of facilities as previously described, is consistent with values considered in the literature for the energy spent for switching on a RSU (Yu et al., 2014).

Table 7.2: Problem Parameters

\bar{S}	\bar{t}	S^{min}	t^{max}	α_1	α_2	w_1	w_2	γ	E^{oh}
10 [Mb/s]	0.5 [s]	0.5 [Mb/s]	10 [s]	$1.3 \cdot 10^{-3}$	10^{-6}	0.5	0.5	{10, 30, 50}	0.13[W · s]

For testing our proposed solution method, we have set 128 available threads, i.e., the number of parallel runs. For considering the case in which the heuristic is completely greedy, i.e., when $\beta = 1$, we have added one more thread for this particular case during the execution, resulting in 129 threads. This avoids the algorithm performing worse than the original greedy deterministic heuristic.

As introduced in Section 7.1.2.1, a new parameter, γ , must be set for the UFLP in the context of IoV. Additionally, we have solved the corresponding model through the Gurobi optimizer in order to generate optimal solutions for small-sized instances and provide a proper performance comparison between the AO and Gurobi. Because of this limitation,

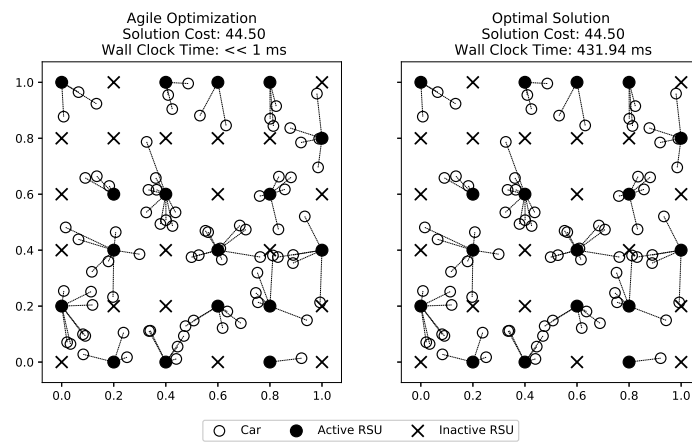
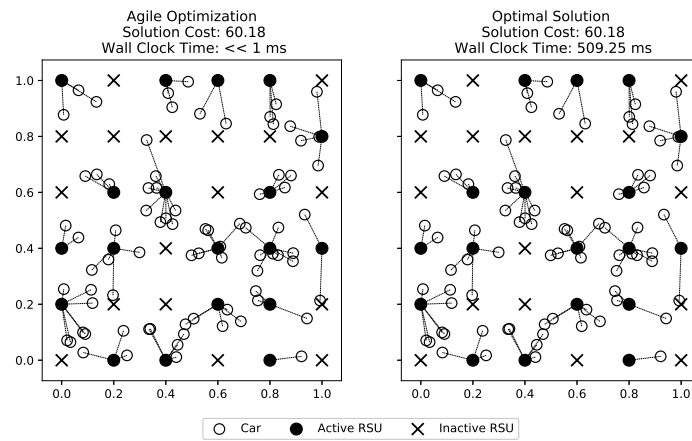
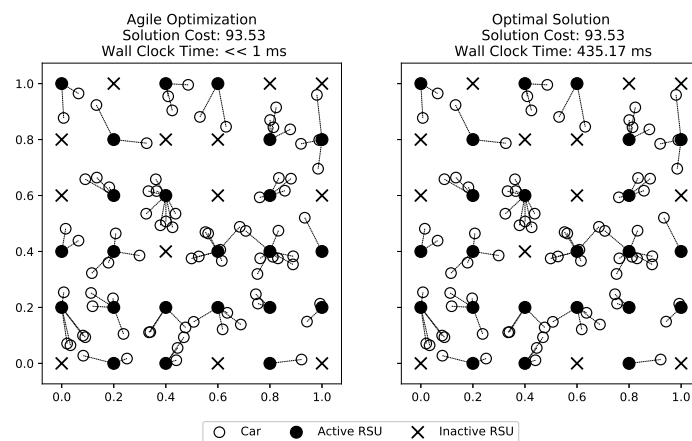
Table 7.3: Comparison of the results obtained by the proposed methodologies.

Instance	Gurobi (1)		De Armas et al. (2017) (2)		AO (3)			gap		
	Optimal	Time	Cost	Time	Cost	Avg. Cost	Time	(2-1)	(3-1)	(3-2)
500-10	798,577	15.54	816,911	0.06	811,769	825,083.54	0.06	2.3%	1.7%	-0.6%
500-100	326,790	12.69	334,182	0.01	332,657	334,433.03	0.01	2.3%	1.8%	-0.5%
500-1000	99,169	15.12	99,630	0.02	99,512	99,645.53	0.02	0.5%	0.3%	-0.1%
1000-10	1,434,154	516.45	1,496,913	0.03	1,476,219	1,496,014.28	0.07	4.4%	2.9%	-1.4%
1000-100	607,878	117.60	623,351	0.03	621,723	623,898.95	0.05	2.5%	2.3%	-0.3%
1000-1000	220,560	84.18	225,232	0.03	224,811	225,290.53	0.04	2.1%	1.9%	-0.2%
1500-10	2,000,801	12,109.82	2,081,386	0.08	2,065,163	2,092,176.60	0.14	4.0%	3.2%	-0.8%
1500-100	866,454	286.26	900,077	0.08	896,471	900,586.16	0.10	3.9%	3.5%	-0.4%
1500-1000	334,962	211.74	344,279	0.07	343,719	344,272.48	0.07	2.8%	2.6%	-0.2%
2000-10	2,558,118	2,407.97	2,683,346	0.15	2,655,188	2,687,506.16	0.21	4.9%	3.8%	-1.0%
2000-100	1,122,748	488.35	1,166,994	0.14	1,161,132	1,165,985.98	0.23	3.9%	3.4%	-0.5%
2000-1000	437,686	419.54	450,549	0.15	449,843	450,552.88	0.22	2.9%	2.8%	-0.2%
2500-10	3,099,907	240,588.51	3,223,279	0.50	3,191,427	3,231,097.21	0.50	4.0%	3.0%	-1.0%
2500-100	1,347,516	1,534.58	1,398,526	0.23	1,392,504	1,397,435.05	0.48	3.8%	3.3%	-0.4%
2500-1000	534,405	758.51	547,825	0.46	546,985	548,023.74	0.49	2.5%	2.4%	-0.2%
3000-10	3,570,766	2,960.90	3,714,590	0.61	3,684,684	3,736,407.78	0.76	4.0%	3.2%	-0.8%
3000-100	1,602,154	8,873.80	1,653,616	0.72	1,649,793	1,654,669.25	0.72	3.2%	3.0%	-0.2%
3000-1000	643,463	1,692.01	660,541	0.69	659,298	660,499.33	0.69	2.7%	2.5%	-0.2%
Average	1,200,339	15,171.87	1,245,624	0.23	1,236,828	1,248,532.14	0.27	3.1%	2.6%	-0.5%

regarding the size of instances, three different solutions of allocating 50 cars to 16 RSUs are depicted in Figure 7.6, which refers to assigning different weights to the QoS and Energies by changing the γ value. Since this problem instance is considered of small size, it could be optimally solved by the Gurobi.

As we can notice, our method was able to reach the optimal solutions for the tested problem instances. Also, we can see that by increasing γ the quality of service is given more importance, hence more RSUs will be active in the solution. It's important to underline that by the very nature of this agile approach, the parameters can be changed for every execution, hence allowing more user control on the system's consumption and reliability. Our solution method was able to achieve speedups of many orders of magnitude over the Gurobi solver on the same test instances.

In order to have a fair comparison with other benchmarks in the literature, and for considering extensive results that cannot be obtained if considering a realistic IoV environment, in the following, we focused on a particular case of the IoVP when the cost between vehicles and RSUs is supposed to be driven by the distance between them. Consequently, this mainly impacts the QoS parameters. Moreover, the energy consumption of activating an RSU corresponds to the cost of opening it, and the γ is set to 1. This simplification allows reducing the problem to the traditional UFLP. Therefore, in Table 7.3, the results obtained by setting the parameters as described are presented. For each instance, it is presented: (i) the optimal solution obtained through Gurobi solver and its running time; (ii) the solution obtained by De Armas et al. (2017)'s deterministic heuristic and its running time in our experimental environment; (iii) the best-found solution generated by our method, the average solution cost and its processing time. Later, the gaps among the mentioned solving methods are presented.

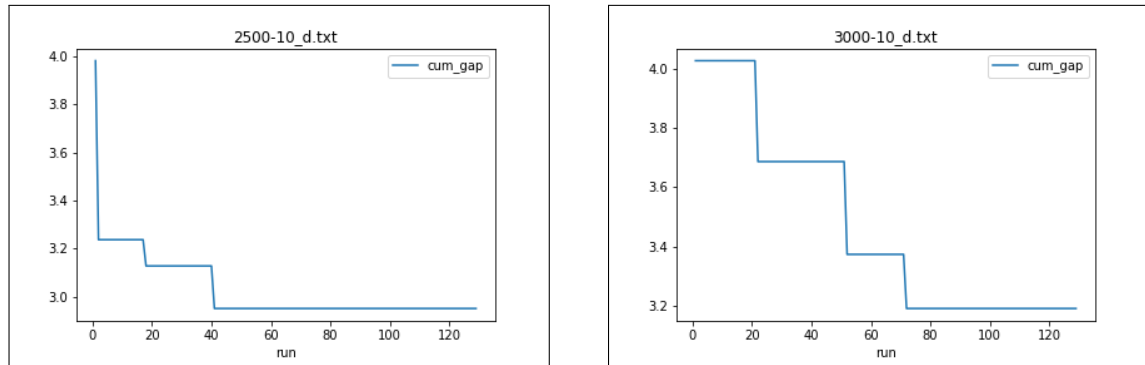
(a) $\gamma = 10$.(b) $\gamma = 30$.(c) $\gamma = 50$.Figure 7.6: The same instance is solved three times with different values of γ .

When analyzing the results, we can notice that our AO approach is able to improve previously published results in the literature, which can be considered compatible with the IoV scenario, when setting the parameters accordingly. By embedding our BR version of the existing heuristic into our AO framework, we are able to reduce around 1% the cost of the solutions when comparing to solving method 2. Apart from this average improvement, all solutions generated by our AO mechanism have a lower cost when compared with method 2, resulting in negative gap values as reported in column *gap* (3-2). On the other hand, it is also noticeable that the method 2 was able to generate near-optimal solutions for all benchmark instances. For this reason, even being able to beat this solution method, our approach presented tiny improvements since method 2 was efficient enough for solving this problem. When comparing our method with optimal solutions, we are only 2.6% from optimal values, on average, but requiring noticeably less computational time, in order of milliseconds. This particularity highlights the power of our AO approach in environments where re-optimization should be frequently done as the system changes.

It is worth to be noticed that 3GPP standardization bodies have defined several Internet of Vehicle services, with different requirements (Technical Specification Group Services and System Aspects, 2019; Technical Specification Group Services and System Aspects, 2020). While advanced and remote driving services require very stringent latency requirements, in the order of a few milliseconds, other services, such as reporting, generic Vehicle to Network communications, do need latency requirements in the order of seconds. Within this context, different instances effectively represent different deployment scenarios. Hence, low latency services are preferred to be mapped on small instances (i.e., managed as smaller populations) while services with no stringent latency can be mapped on larger instances (i.e., managed as bigger populations).

In Figure 7.7, we present the convergence of the best-found solutions according to the increase in the number of threads. As mentioned, our solution method lies in multiple executions of a BR algorithm. By plotting the cumulative minimum of the solutions given by these threads, we can determine a good trade-off between the number of required threads (i.e., computing power) and the overall improvement over the deterministic solution. In Figures 7.7a and 7.7b, this convergence is presented for problem instances 2500-10 and 3000-10, respectively. Since the proposed algorithm is based on an efficient method, the embedding of a biased-randomized heuristic was also able to produce solutions within a gap of 1%. As expected, the more computational power is available, i.e., the number of threads, the more efficient the method is.

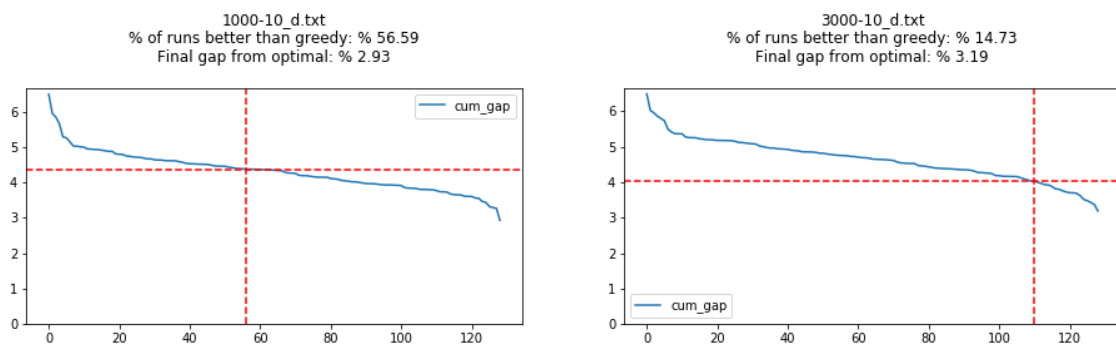
Figure 7.8 depicts the improvement rate of the 129 generated solutions during our AO lifecycle, in terms of the gap. In this case, we compare our results with De Armas et al. (2017)'s heuristic (the red cross intersection), which represents the solution obtained by the deterministic heuristic. As we can see, for problem instance 1000-10 (Figure 7.8a), about 57% of the generated solutions are better than the deterministic heuristic result. On the other hand, for problem instance 3000-10 (Figure 7.8b), which is of higher magnitude, only 15% of the generated solutions were able to beat the De Armas et al. (2017)'s heuristic.



(a) Convergence of solutions of instance 2500-10.

(b) Convergence of solutions of instance 3000-10.

Figure 7.7: Results obtained when varying the number of parallel runs.



(a) Convergence of solutions of instance 1000-10.

(b) Convergence of solutions of instance 3000-10.

Figure 7.8: Results obtained when varying the number of parallel runs, including the improvement rate of the solutions when comparing with De Armas et al. (2017)'s heuristic, given by the red cross intersection.

7.1.4.3 Results for the Multi-Period Internet of Vehicles Problem

The first set of computational experiments relies on measuring the performance of our approaches when incorporating the demands shifting model into the MPIoVP. As introduced in Section 7.1.3.2.1, the savings-based heuristic has been tested into different solution frameworks. For any of the applicable cases, the value of parameter β is selected between $\beta \in [0.4, 0.8]$, as described in the previous section. Since the *single-operative* and *all-operative* static approaches consider a single operative facility and all the facilities operative over time, respectively, they do not include the savings-based heuristic as a guide.

Table 7.4 shows the computational results for the five different scenarios. At this stage, the planning horizon is set to 20 periods, i.e., $T = 20$. For each test instance and scenario, we can see: the average objective function value (OF) of all periods, the average number of operative RSUs ($|\mathcal{R}|$) across all periods, and the average computational time required to generate the results, in seconds. The average OF cost was divided by $1e6$ in order to provide readers with more readable results. The ‘gap’ columns present the comparative performance of the different scenarios according to their respective average OF values.

In Table 7.4, we consider the *keep-the-first* strategy as the ‘naive’ approach against which other scenarios are compared. In this approach, the system is optimized at the beginning of the planning horizon, and the resulting configuration of operative RSUs and assigned vehicles is kept for the remaining periods. Due to the high variability of the switching on cost, given by each scheme, it is convenient to analyze the results accordingly.

When activating facilities is expensive (i.e., instances with the suffix -10), it is noticeable how costly the *all-on* approach is when compared with the *keep-the-first* strategy –see column *gap* (3)-(2). It is true that, by activating all the RSUs in the system, the assignment cost becomes null, since each node is assigned to itself. However, there are still demands to be supplied, which incurs additional variable costs into the system. Since the activation cost is too large in this scenario, the higher the difference between the number of operative facilities is, the higher the gap among the respective instances will be (e.g., instance 3000 – 10). By analyzing the performance of both *dynamic* approaches against the *keep-the-first* strategy, it is clear that both of them outperform this naive approach, being the *fully-dynamic* strategy of better overall performance. This behavior is expected since the system is re-optimized in each period.

Another extreme scenario regards the case in which the activation cost of RSUs is cheap (i.e., instances with the suffix -1000). In this case, there are no significant differences among the approaches, since the number of open RSUs exceeds 95% of all facilities in all cases. Hence, the main differences among the cost of these strategies rely on how the assignment of nodes, which are not activated as facilities, is set. Particularly, the *keep-the-first* performs slightly better than the *all-on* approach, since a fraction of the fixed cost can be saved for not opening a few RSUs –and, then, the assignment of those facilities that are not activated does not imply expensive variable costs. Regarding the dynamic approaches, they present similar performance, since the majority of their facilities are operative, which generates minimal differences in their variable costs.

Table 7.4: Computational results in terms of cost, facilities and completion time for the different scenarios.

instance	single-on (1)		all-on (2)		keep-the-first (3)		partially-dynamic (4)		fully-dynamic (5)		gap									
	OF	avg. $ \mathcal{R} $ time	OF	avg. $ \mathcal{R} $ time	OF	avg. $ \mathcal{R} $ time	OF	avg. $ \mathcal{R} $ time	OF	avg. $ \mathcal{R} $ time	(3)-(1)	(3)-(2)	(3)-(4)	(3)-(5)	(4)-(5)					
500-10	148.2	1	0.0	5.9	500	0.0	5.9	206	0.0	5.3	241	0.0	5.2	245	0.1	2,430.1%	0.6%	-9.8%	-11.0%	-1.3%
1000-10	287.9	1	0.0	16.7	1,000	0.0	12.5	239	0.0	11.6	288	0.1	11.5	293	0.2	2,205.1%	33.3%	-7.0%	-7.9%	-0.9%
1500-10	343.6	1	0.0	30.6	1,500	0.0	18.9	259	0.0	17.9	310	0.2	17.7	315	0.5	1,717.7%	61.7%	-5.4%	-6.2%	-0.8%
2000-10	509.3	1	0.0	47.1	2,000	0.0	24.6	283	0.0	23.6	328	0.4	23.5	335	0.9	1,970.8%	91.3%	-4.0%	-4.6%	-0.6%
2500-10	510.3	1	0.0	65.8	2,500	0.0	29.9	307	0.0	28.6	362	0.7	28.5	366	1.3	1,607.0%	119.9%	-4.2%	-4.6%	-0.5%
3000-10	594.1	1	0.0	86.4	3,000	0.0	35.7	329	0.1	34.6	373	0.8	34.4	381	1.7	1,566.1%	142.3%	-3.0%	-3.4%	-0.5%
500-100	148.2	1	0.0	0.6	500	0.0	0.6	496	0.0	0.6	491	0.0	0.6	491	0.0	24,300.1%	0.8%	0.2%	0.1%	-0.1%
1000-100	287.9	1	0.0	1.7	1,000	0.0	1.7	944	0.0	1.7	929	0.1	1.7	928	0.2	16,384.0%	-2.1%	-1.7%	-2.2%	-0.4%
1500-100	343.6	1	0.0	3.1	1,500	0.0	3.2	1,299	0.0	3.1	1,305	0.2	3.1	1,306	0.5	10,626.9%	-2.4%	-2.7%	-3.4%	-0.7%
2000-100	509.3	1	0.0	4.8	2,000	0.0	4.9	1,628	0.0	4.7	1,599	0.4	4.7	1,597	0.9	10,313.3%	-1.9%	-3.1%	-3.8%	-0.8%
2500-100	510.3	1	0.0	6.7	2,500	0.0	7.1	1,724	0.0	6.6	1,786	0.7	6.5	1,788	1.4	7,126.9%	-5.3%	-7.2%	-8.1%	-1.1%
3000-100	594.1	1	0.0	8.8	3,000	0.0	9.0	1,981	0.0	8.5	1,997	1.0	8.4	1,999	2.3	6,509.7%	-2.4%	-5.4%	-6.3%	-1.0%
500-1000	148.2	1	0.0	0.1	500	0.0	0.1	496	0.0	0.1	496	0.0	0.1	496	0.0	177,711.6%	0.6%	0.0%	0.0%	0.0%
1000-1000	287.9	1	0.0	0.2	1,000	0.0	0.2	993	0.0	0.2	993	0.1	0.2	993	0.2	133,984.2%	0.5%	0.0%	0.0%	0.0%
1500-1000	343.6	1	0.0	0.4	1,500	0.0	0.4	1,468	0.0	0.4	1,468	0.2	0.4	1,468	0.4	91,947.2%	1.7%	0.0%	0.0%	0.0%
2000-1000	509.3	1	0.0	0.6	2,000	0.0	0.6	1,953	0.0	0.6	1,952	0.4	0.6	1,952	0.8	91,111.6%	2.0%	0.0%	0.0%	0.0%
2500-1000	510.3	1	0.0	0.8	2,500	0.0	0.8	2,414	0.0	0.8	2,413	0.7	0.8	2,413	1.3	67,159.5%	3.0%	0.0%	0.0%	0.0%
3000-1000	594.1	1	0.0	1.0	3,000	0.0	1.0	2,899	0.0	1.0	2,896	0.8	1.0	2,896	1.8	60,274.5%	3.0%	0.0%	0.0%	0.0%
Average	-	-	0.0	-	-	0.0	-	-	0.0	-	-	0.4	-	-	0.8	39,385.9%	24.8%	-2.9%	-3.4%	-0.5%

The last scenario refers to the case in which the activation cost is moderate (i.e., the instances with the suffix -100). Unlike previous scenarios, the *all-on* approach presents better

performance, on average, than the *keep-the-first* strategy. Since the cost of activating a facility is not too expensive, a large number of facilities is frequently deployed. However, there are some cases in which the difference between the number of potential facilities to be activated and the number of already operative facilities is reasonably large (e.g., instances 2000-, 2500-, and 3000-100). In these cases, the cost is mainly generated by the many assignments, implying higher variable costs. Therefore, for this costing scheme, the magnitude of both the activation and assignment costs results in better performance for the *all-on* strategy against the *keep-the-first*. Nevertheless, when analyzing the performance of the dynamic approaches, both of them present better average performance than the remaining strategies.

Considering the complete benchmark instances –from the different activation schemes– we can assert, from column *gap* (3)-(1) to *gap* (3)-(5), that the *fully-dynamic* approach presents best overall performance. Since this strategy re-optimizes the solution in every period, it is expected to be more efficient. However, it is noticeable how its efficiency worsens as the activation cost of facilities decreases. Regarding computational times, the computation time required by the *fully-dynamic* approach is still low, being only 0.8 seconds slower than the *keep-the-first* strategy. When comparing our two dynamic approaches, in column *gap* (4)-(5), the *fully-dynamic* strategy outperforms the *partially-dynamic* in 0.5%, on average. However, it requires 0.4 additional seconds to obtain these results. Figure 7.9 presents, for all solving approaches, a box-plot of the corresponding results. The *single-on* strategy is not included due to its numerical scale being too high.

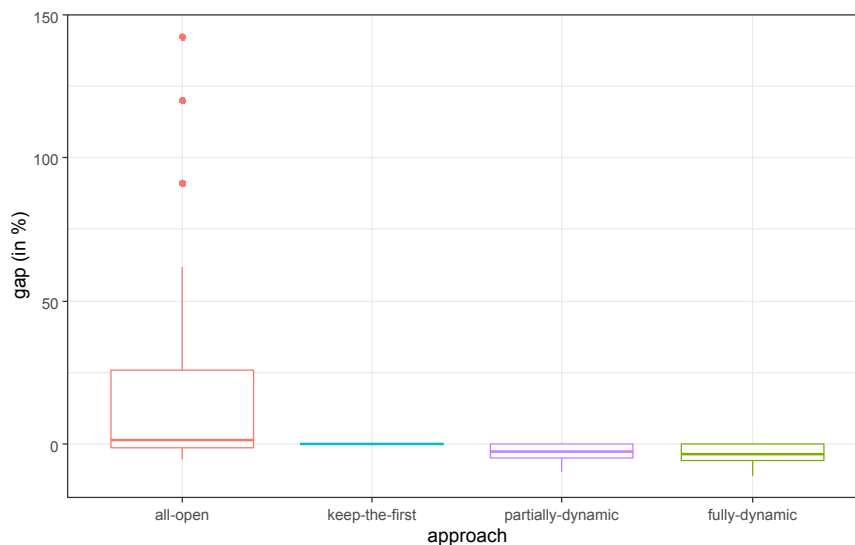


Figure 7.9: The variability of the average results for each scenario, in terms of gap.

As we can notice in Figure 7.9, the *all-on* strategy is clearly inefficient, mainly when the activation cost of facilities is expensive. Moreover, we can notice that both the *fully* and *partially-dynamic* approaches are able to generate better results than the *keep-the-first*. However, we cannot assert which method is statistically better only by analyzing this set of box-plots. Therefore, an ANOVA and Tukey test were performed to identify significant

differences among the approaches. With a p – value = 0.00105, the ANOVA test guarantees that there are statistical differences among the approaches. By analyzing the results provided by the Tukey test, we can assert that there is no statistical difference among the *keep-the-first*, *partially*, and *fully-dynamic* approaches when considering all the test instances. However, these approaches are statistically better than the *keep-the-first* strategy, in general. Therefore, for this set of benchmark instances, the *fully-dynamic* is considered the best approach to cope with the MPIoVP since it performs statistically similar to the previous two approaches but it is able to generate better results on average.

For the problem instance composed of 500 cells, Figure 7.10 presents the convergence of the OF value over the periods. Similar to Table 7.4, the comparisons are performed by considering the *keep-the-first* as the reference approach.

As we can see in Figure 7.10a –which represents the results obtained for the expensive activation scenario–, the *fully-dynamic* approach does not only provides better overall performance, but it is also able to generate better configurations for all the periods over time, achieving up to 17% of improvement in periods 18 and 20. Similarly, when comparing the *partially-dynamic* with the *keep-the-first* approaches, the same behavior is noticed, with up to 16% of improvement in periods 18 and 20. These conclusions support the efficiency of the dynamic strategies when demands vary over time. On the other hand, by analyzing Figure 7.10c one can conclude that, for the cheapest activation scenario, the approaches' performance does not present any significant difference since almost all the facilities are deployed. Nevertheless, it is noticeable that the *fully-dynamic* approach presents better performance, mainly in the last period of the planning horizon. Finally, when analyzing the results under moderate activation cost (Figure 7.10b), one can observe that the *dynamic* strategies are able to properly react to the system changes, according to the demands shift.

7.1.5 Conclusions

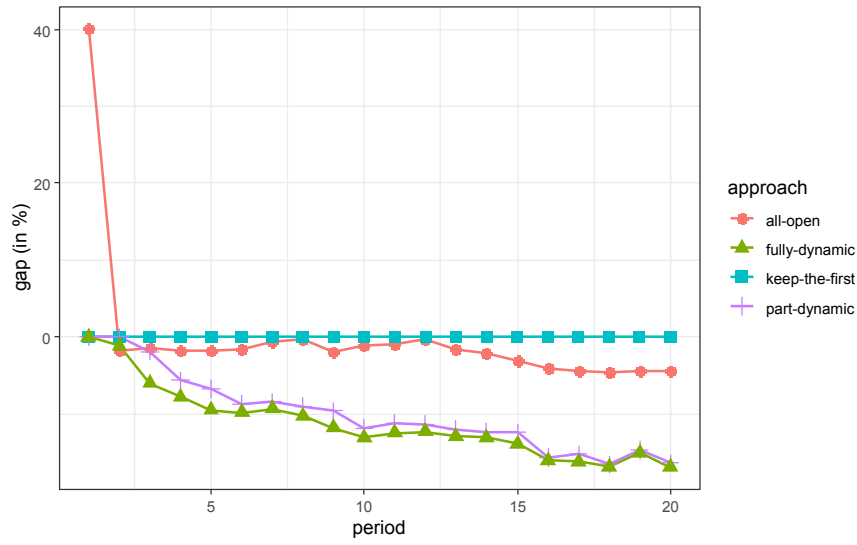
In this section, an agile optimization framework based on the composition of multiple biased-randomized runs of an existing heuristic has been developed for solving a rich version of the classical uncapacitated facility location problem. Differently from logistics and supply chain, where decisions are difficult to reverse and time is not a constraint for generating feasible solutions, we addressed the UFLP in the context of telecommunication systems and smart cities, where the goal is to optimize the configuration of the RSUs in each period, as a response to the time-evolving vehicles' communication needs. In this case, this dynamic environment must be re-optimized as customer demands change over time.

For doing that, an existing heuristic was extended by introducing a random component into its solution space exploring process, controlled by a single parameter β . The resulting biased-randomized heuristic is able to produce different solutions that can be better than the original one by regulating the "greediness" of the algorithm by tuning a factor $\beta \in [0, 1]$. By exploiting the high parallelization capabilities of modern hardware, multiple independent runs can be performed without varying the effective wall clock time of the computation. Therefore, as a next step, this extended heuristic has been embedded into the AO framework for running several executions in parallel, then not requiring any extra computational time

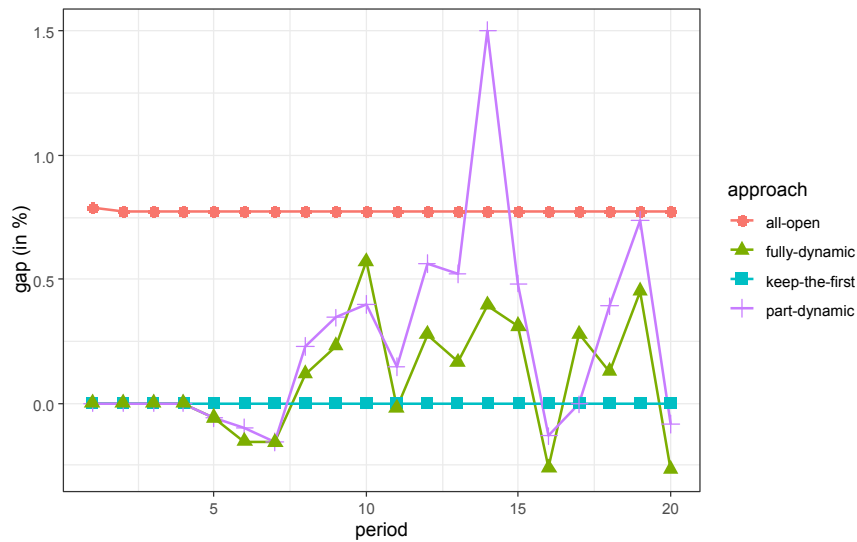
as the deterministic heuristic. We simulate this parallelization by running and timing each different execution in a sequential manner.

As results show, our method is able to obtain identical solutions as those generated by a commercial solver for the IoVP, and improve all the instances solutions generated by an existing and competitive solution method in the literature. Apart from being able to beat this alternative heuristic, only 1% of improvement was achieved. This can be explained by the reason the alternative heuristic is good enough to provide near-optimal solutions. Regarding the MPIoVP, numerical experiments carried out have illustrated the benefits of the proposed agile optimization approach, which is dynamic, over other more static approaches. Our results show how significant cost reductions can be obtained by adopting an agile optimization methodology as the one presented here.

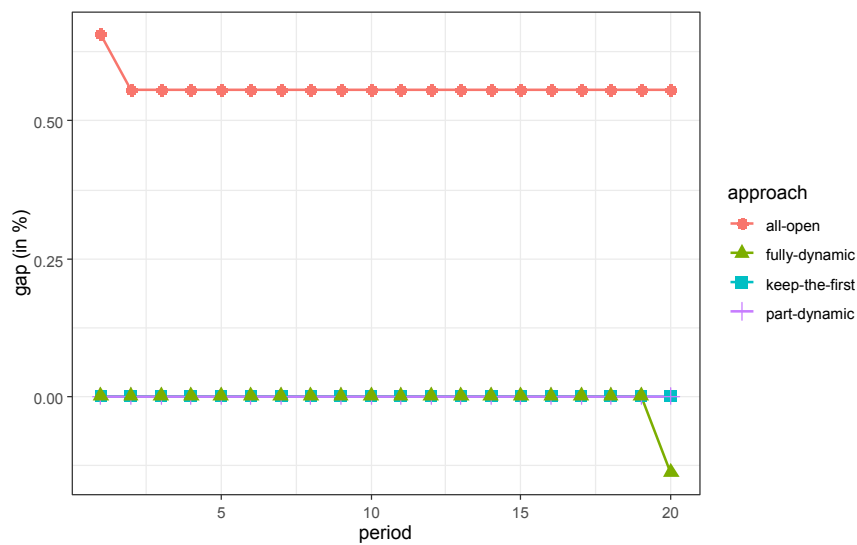
Regarding the use of our AO framework for solving this class of dynamic problems, we must highlight that, in real-life, unlike this work, a graphic card processor (GPU) –with enough resources– could be used to avoid the changes of context and synchronization overheads of a common CPU. Consequently, the sort of problems in which available resources are not enough for running all these threads in parallel without getting possible overheads can be discarded. Notice that a GPU has created a new efficient and effective way to do massive parallel computing, due to it contains hundreds or thousands of cores. In other words, the lack of resources can destroy the idea behind agile optimization, in which the performance -in terms of execution time- is degraded as the number of threads increases, due to changes of context and the use of the resources.



(a) Instance 500-10 (expensive activation cost).



(b) Instance 500-100 (moderate opening cost).



(c) Instance 500-1000 (cheap opening cost).

Figure 7.10: The convergence of the OF value over the periods, in terms of gap, for problem instance 500.

Chapter 8

Applications in Healthcare

This chapter ¹ describes a case study regarding the use of ‘agile’ computational intelligence for supporting logistics in Barcelona’s hospitals during the COVID-19 crisis in 2020. Due to the lack of sanitary protection equipment, hundreds of volunteers, the so-called “Coronavirus Makers” community, used their home 3D printers to produce sanitary components, such as face covers and masks, which protect doctors, nurses, patients, and other civil servants from the virus. However, an important challenge arose: how to organize the daily collection of these items from individual homes, so they could be transported to the assembling centers and, later, distributed to the different hospitals in the area. For over one month, we have designed daily routing plans to pick up the maximum number of items in a limited time –thus reducing the drivers’ exposure to the virus. Since the problem characteristics were different each day, a series of computational intelligence algorithms were employed. Most of them included flexible heuristic-based approaches and biased-randomized algorithms, which were capable of generating, in a few minutes, feasible and high-quality solutions to quite complex and realistic optimization problems. This chapter describes the process of adapting several of our ‘heavy’ route-optimization algorithms from the scientific literature into ‘agile’ ones, which were able to cope with the dynamic daily conditions of real-life routing problems. Moreover, it also discusses some of the computational aspects of the employed algorithms along with several computational experiments and presents a series of best practices that we were able to learn during this intensive experience.

8.1 The Application of Biased-randomized Algorithms for Supporting Logistics Decisions During the COVID-19 Crisis in 2020

The COVID-19 pandemic crisis is one of the toughest global challenges we have faced in decades. Several leaders have even stated it as “the biggest one since the Second World War”. The exponential growth of cases that needed medical attention led to a sudden shortage of protective material so that the medical and support staff were subject to a high risk to also become infected, endangering the needed level of attention in hospitals and also a

¹The contents of this chapter are based on the following work:

- Tordecilla, R.; **Martins, L. C.**; Saiz, M.; Copado-Méndez, P. J.; Panadero, J.; Juan, A. A. (2021): [Agile Computational Intelligence for supporting Hospital Logistics during the COVID-19 Crisis](#). *Computational Management*, 383-407.

faster spread of COVID-19. By March 2020, the pandemic had a strong impact on many countries, such as Italy and Spain. As it happened in other regions, in the metropolitan area of Barcelona, a community of volunteers, the so-called “Coronavirus Makers” Community, arose intending to supply protective material to the staff working in the hospitals, nursing homes, and emergency medical attention. The main tool used was home 3D-printers, which helped to iterate the design very fast in order to reach, within a few days, the design level that was considered acceptable by the staff in charge of guaranteeing safety and quality of the produced items (Figure 8.1). It soon became noticeable that the bottleneck was the logistic side of this endeavor because the lock-down situation meant that each 3D printer was located at each individual home, and route planning needed technological support, in order not to expose the drivers to more risk than strictly necessary.



Figure 8.1: The main goal of the Makers’ community was to supply protective items to hospitals and healthcare centers.

This chapter discusses the experience of matching several professional and personal profiles that typically work with very different approaches, because of the nature of their work: scientists, volunteers, makers, and entrepreneurs. In this case, it was needed to find a fast way of applying the knowledge gathered within years of research to an urgent need, where every day counts. The target was to support the makers’ community (with each maker located in his / her individual home) on their voluntary initiative to supply the sanitary staff with as much protective material as possible and with a limited time to avoid unnecessary exposure for the drivers.

The main contributions of this chapter are: (i) it describes a real-life case in which computational intelligence was used to support hospital logistics during the COVID-19 pandemic crisis; (ii) it illustrates how real-life logistics might be rich in the sense that they combine multiple routing problems with dynamic characteristics and constraints –which might vary from day to day; (iii) it provides an example of how ‘agile’ optimization can be applied –in combination with other technologies– to support decision making in scenarios under stress;

and (iv) it discusses how to develop new agile-optimization tools that can efficiently cope with the aforementioned scenarios.

8.1.1 The Barcelona's Makers Community

The 'Makers' community was born to contribute with creative capacity and offer a service to the healthcare system, the geriatric staff, and home-support personnel. This was a 100% altruistic and non-profitable initiative. The aim was to alleviate the need for additional protective material in hospitals and health centers derived from the scarcity of resources due to the unprecedented level of demand worldwide generated by the COVID-19 outbreak. The initiative was conceived in less than 48 hours between March 12th and 14th, and grew at an average rate of 1,000 new volunteers per day during the first two weeks. The makers' community from Barcelona and the surrounding provinces adhered soon to this initiative, and the community was already handing out material to the hospitals on March 16th, 2020. By the end of that same week, the Barcelona community had grown significantly, leading to communication to their community on March 21st informing that the next routes will only collect material for makers that have a minimum amount of parts manufactured –although, in those critical times, every shield counted in order to protect the sanitary staff. Figure 8.2 provides an example of the problem magnitude in the area of Barcelona, for a specific crisis period. As we can notice, several pick-up and delivery points are geographically distributed, being the coordination of both loading and unloading activities the next challenge to be faced. This was the first sign that the logistics were becoming a bottleneck and further help was needed.

Figure 8.3 depicts the overall process. Firstly, every maker is printing the cap through a 3D printer, and then the caps are disinfected by the maker. Due to the lock-down restrictions, the makers cannot (and must not) transport the caps themselves, so there is a transport arranged by the node coordinators to pick up all material and hand it out to the person being responsible for that area. Then, the node team is assembling the finished product (3D-printed cap with an acetate shield and a head-adjusting elastic band). Afterward, the finished product is delivered to the medical centers, which complete the second disinfection round, typically via ozone sterilization.

The proposal to collaborate with the research team was made on March 23rd. The work was done in one-day sprints, setting the highly ambitious target to be able to have route proposals following the adapted algorithms by the next day at 9 a.m. or before. The first sprint was almost successful. For a few minutes, it could not be implemented due to several challenges in the database interfaces and synchronization between all planners involved. It became successful the day after, and from that day on, it was possible to propose high-quality routes to the drivers. The route proposal is handed out to the drivers via a native file that can be opened in a popular mapping application of common use in cell phones.

A process was established in order to improve the synchronization between all the volunteers involved in the node planning and the research team (Figure 8.4). In this specific case, transport 1 (pick up at makers' homes) and transport 2 (delivery at sanitary centers) were planned and executed together to gain further advantage from the algorithms already



Figure 8.2: Some of the pick-up locations geographically distributed in the area of Barcelona.

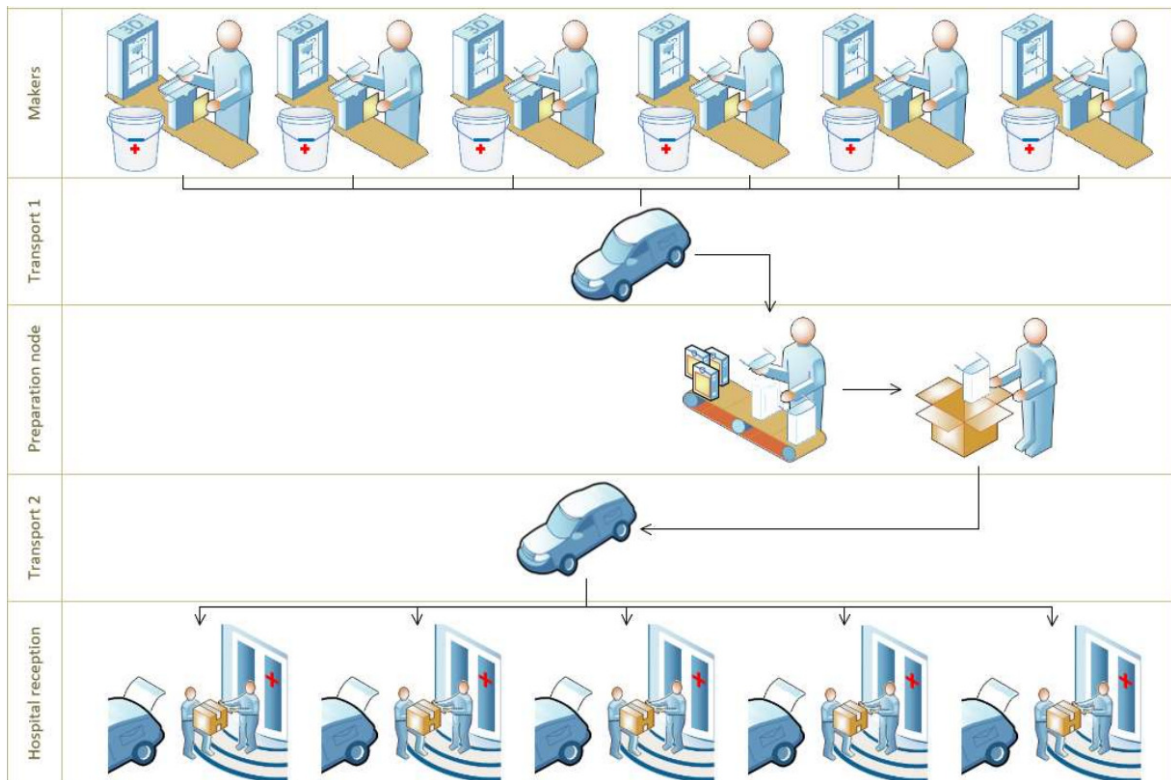


Figure 8.3: Overview of the material flow.

in place. The makers' community continued to work on this initiative for over two months (March, April, and part of May), reaching a total of 75,000 or more face shields, above 11,000 door openers, and over 53,000 ear savers through all nodes established in the metropolitan area of Barcelona.

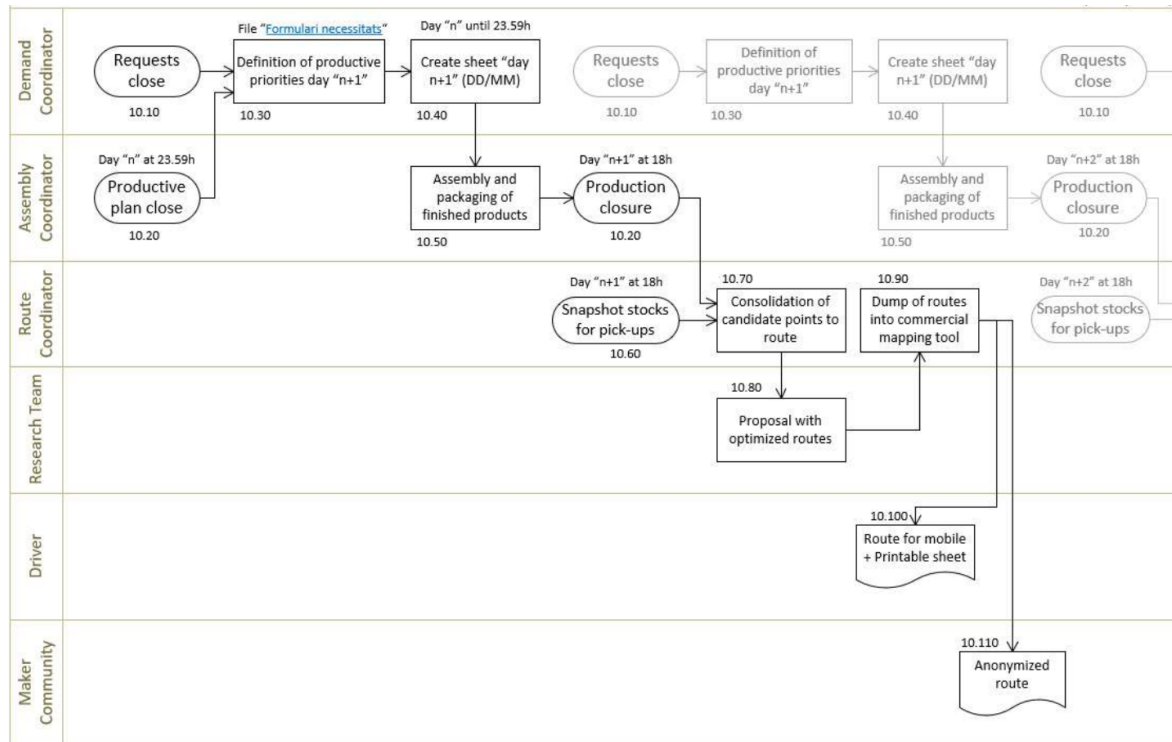


Figure 8.4: Daily node planning and interaction with research team.

8.1.2 Routes Generation Process

The "Proposal with optimized routes" step in Figure 8.4 is the process in which routes are generated through the use of agile computational algorithms. This process is showed extensively in Figure 8.5. Initially, details about each point to visit are provided in a spreadsheet by the Route coordinator, namely: quantity of medical supplies to pick up, municipality, address, postal code, geographic coordinates, and type of node (origin, destination, or mandatory node). As presented in Figure 8.3, demand nodes can either offer (makers and preparation nodes) or consume (hospitals and preparation nodes) medical supplies. Nevertheless, consumption points do not have an established demand, since they consume all supplies offered by the makers. Therefore, data for demand are only given by pick-up points.

The input data is then analyzed by the research team to identify the characteristics of each instance. This step is necessary since such requirements are set every day, posing a new challenge for the team. Details about these requirements are provided in Section 8.1.7. In general, our hospital-logistics problem shows characteristics of several rich variants of the vehicle routing problem (VRP). Nevertheless, we address all daily challenges as variants of two big groups: (i) a VRP-related challenge, in which all customers (makers' houses) are visited and the objective is to minimize the total time requested in completing the collection;

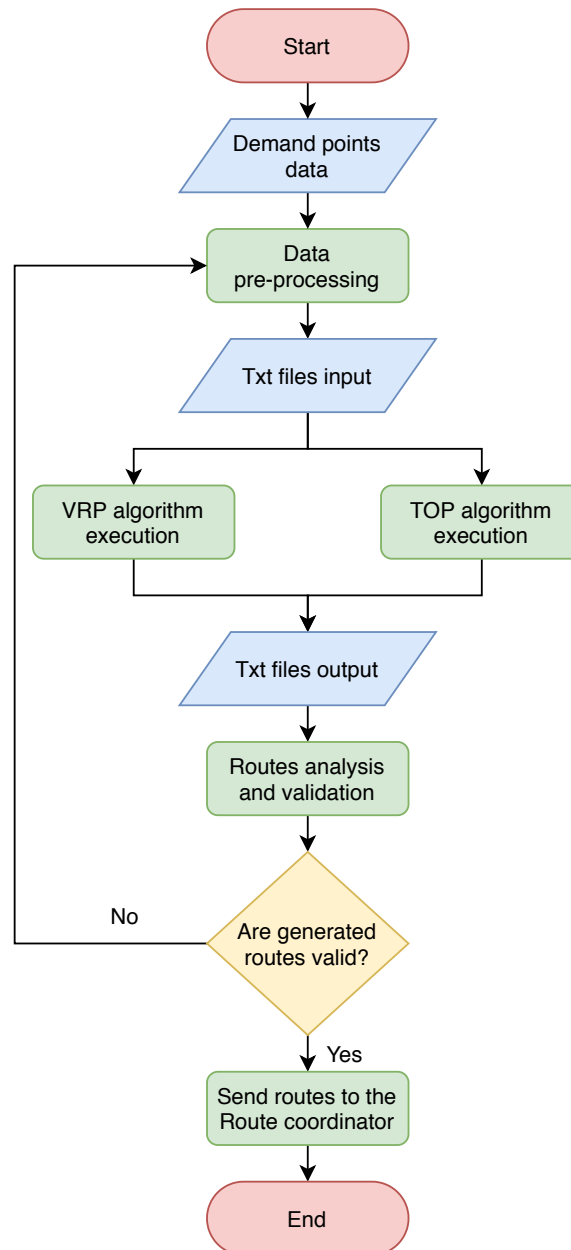


Figure 8.5: Flowchart of the routes generation process.

and (ii) a rich team orienteering problem (TOP), in which a time limit must be met and, hence, not all customers can be visited –since the number of drivers and pick-up vehicles is also limited. Therefore, some customers are not visited, seeking a maximum reward while satisfying the constraints. Notice that, since the problem characteristics change every day, the challenge we faced every night was typically not a pure VRP or a pure TOP, but a hybrid between them.

Once the instance characteristics have been identified, the spreadsheet is adjusted and processed to convert it into standardized *txt* files through a Python code. This code generates two files: (i) a nodes-file with details about demand, coordinates, and type of node; and (ii) an edges-file with information about the estimated travel time and the distance between each pair of nodes in the instance. A web mapping service is called by the Python code to

estimate these parameters. Then, VRP and TOP algorithms are executed in parallel. Both algorithms were coded as Java applications. They are able of generating high-quality solutions in a matter of seconds. Each algorithm outputs a standardized *txt* file with the next variables for each designed route: total travel time, total traveled distance, total collected elements, total visited nodes, and nodes sequence to visit. Additionally, the output file highlights the route with the highest travel time, since this variable represents one of the main objectives to be minimized by the algorithms. Next, the research team analyzes the designed routes to guarantee that they are valid according to the instance requirements. Also, routes are depicted using a VBA / Excel application. Both travel times and routes' depiction are used for the validation process. For example, some instances require a node-clustering process, which is performed by the research team in the pre-processing step. However, sometimes generated routes are quite different in terms of total travel time, which should be avoided so that volunteer drivers have similar travel times in each route. Hence, if non-balanced routes are generated by the algorithms, data is pre-processed again and the process is repeated. Finally, when valid routes are obtained, they are sent to the coordinator, who provides them to the drivers.

8.1.3 Addressed Problems and Related Works

The first academic mention of a vehicle routing problem was made by Dantzig and Ramser, 1959, as a generalization of the traveling salesman problem. More than 60 years have passed since then, in which a large set of variants has been identified. Each variant of this problem has shown its relevance both academically –given the algorithmic challenges that they pose– and economically –given their high applicability in real-world problems. A thorough study of these variants, solving methods, and applications is carried out by Toth and Vigo, 2014. More recently, concise research is showed by Sharma et al., 2018. Also, a review centered on VRP-related themes instead of traditional variants is made by Vidal et al., 2019. Similar to our work, the urgency and worrying caused by the spread of COVID-19 around the world have encouraged researchers to recognize the potential of solving these classical combinatorial optimization problems to overcome its related challenges, e.g., Chen et al., 2020, and Pacheco and Laguna, 2020. For instance, the pandemic has increased travel and processing times in vehicle routing activities, given the introduction of labors such as cleaning, disinfection, and wearing personal protection equipment (Nucci, 2021). An Indonesian real-case study addressed by Perdana et al., 2020 shows that the COVID-19 pandemic has also increased uncertainty and complexity in food supply chains.

8.1.3.1 The Vehicle Routing Problem

The VRP is a classical *NP-hard* problem (Lenstra and Kan, 1981b) in which a set of customers has a known demand that must be satisfied by a single depot. This depot has a virtually unlimited capacity, and a fleet of homogeneous capacitated vehicles are employed to deliver the demanded units of product. Each customer must be visited only once. Due to the limited capacity of the cargo vehicles, this situation forces the design of several routes in order to

satisfy all the customers' demands (Figure 8.6). Finally, once the vehicle has delivered all its assigned load, it must return to the depot. The objective is to minimize transportation costs, usually measured in terms of total traveled distance or time (Pisinger and Ropke, 2007).

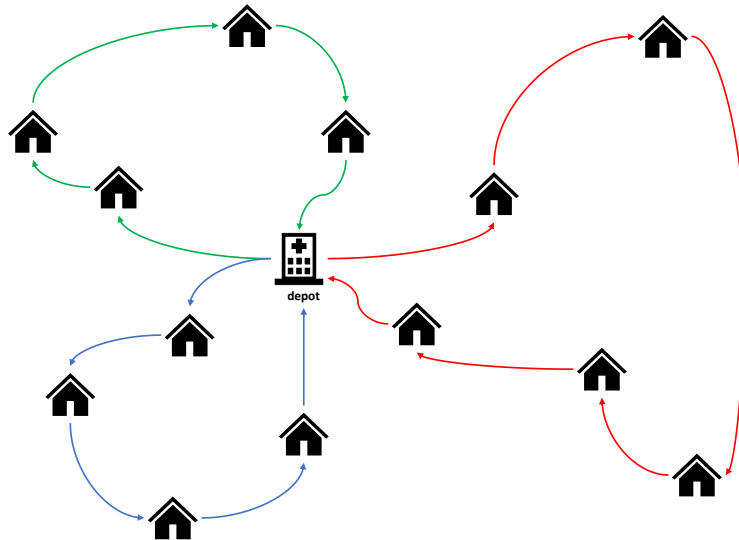


Figure 8.6: An example of a VRP solution with a single central depot.

Real-world problems hardly show characteristics of a pure VRP as described above, since real cases are usually much more complex. This is the case of our hospital logistics problem. For instance, the most common faced variant is related to the possibility that vehicles start and end their routes in different nodes. The relaxation of this constraint defines what we call *open* VRP (OVRP), which, apart from minimizing the travel costs, also aims to minimize the number of used vehicles (Pisinger and Ropke, 2007). From the earlier definition stated by Schrage, 1981, the OVRP has been studied and enriched by many other constraints (Braekers et al., 2016). Among them, we can highlight the rich variants with multiple depots (Lahyani et al., 2019; Brandão, 2020), heterogeneous fleet of vehicles (Yousefikhoshbakht and Dolatnejad, 2017) or even a conjunction of them (Tavakkoli-Moghaddam et al., 2019; Husakou et al., 2020). Although most of these works address a single-objective, there exist multi-objective studies whose aim, for instance, to reduce the total number of routes, the total travel cost, and the longest route altogether (Sánchez-Oro et al., 2020). According to Li et al., 2007, the range of applications that ends up in OVRPs is commonly found in contexts where contractors –who are not employees of the delivery company– use their vehicles and do not return to the depot, such as home delivery of packages and newspapers. In our application context, volunteers start their daily routes from an origin depot (hospitals, health centers, or even their homes) and finish at the end depot (usually, the hospitals or health care centers), where the unloading of the collected goods is performed. Figure 8.7 presents a feasible solution for this variant, in which four routes are designed to serve all customers in order to minimize transportation costs.

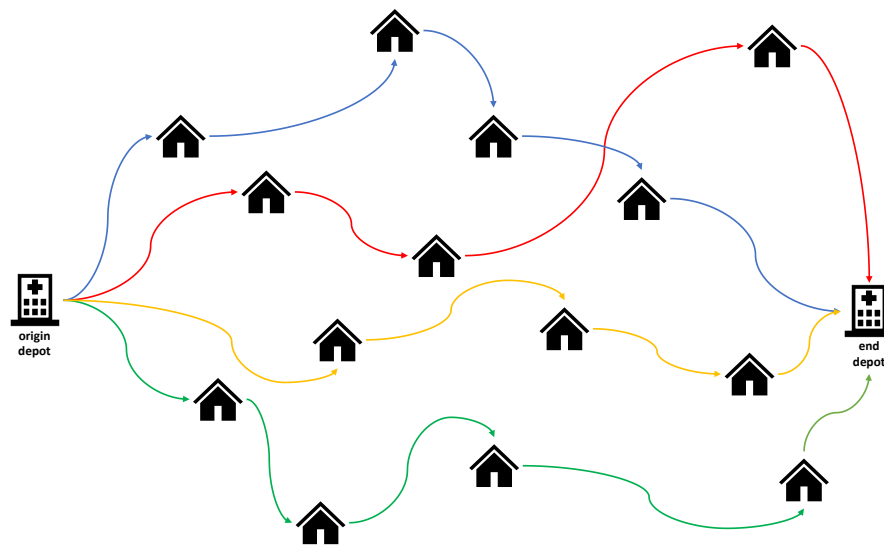


Figure 8.7: An example of a OVRP solution with different start and end depots.

8.1.3.2 The Team Orienteering Problem

In contrast with classical VRPs, which require visiting all customers, there is no obligation regarding this constraint when solving TOP. Besides, the objective is to maximize the collected reward by visiting a set of nodes subject to a set of constraints. It might be the case of many situations in which there are no vehicles (or volunteers) enough for visiting all the pick-up locations from the daily problem data, as faced in this hospital-logistics project. By being an extension of the *NP-hard* orienteering problem (Golden et al., 1987), the TOP is also an *NP-Hard* problem. Similar to the VRP, this problem aims to define a set of vehicle routes in order to optimize one specific objective. In the case of the TOP, the objective is to define a set of vehicle routes such that the total reward collected from visiting each of their assigned nodes, or targets, is maximized (Bayliss et al., 2020a). Each target must be visited once and only by a single vehicle, being each characterized by a positive reward that is collected as soon as it is visited. Given this particularity, the TOP has been ultimately linked to rescue operations (Saeedvand et al., 2020), in which this set of targets –places or tasks– must be chosen to be visited or performed according to their relevance. In our case, the reward is represented by the number of medical supplies to be collected, weighted by their importance at each period from the pandemic crisis. Examples of those materials are face shields, surgical masks, ventilators, handles for opening doors, etc. Therefore, for each pick-up location, its reward is measured by the sum of the weighted available material. Also, we assume that it is not possible to get a partial reward from nodes. The complete route must be performed before a maximum time limit is achieved. Otherwise, the route cannot be properly performed, and all the collected rewards are discarded. Some of its variants still consider precedence constraints (Hanafi et al., 2020), time-windows (Karabulut and Tasgetiren, 2020), stochastic travel times (Panadero et al., 2020c) and dynamic rewards (Reyes-Rubiano et al., 2020), which transforms the TOP into an even harder problem to solve. Figure 8.8 represents

a solution example composed of three routes, being each one of a different color –blue, red, and green– and visiting 2, 3, and 4 collection points, respectively. Besides, notice that the origin and end depots are different, and volunteers do not visit 5 locations due to the limited length of the routes. The nodes which are not considered are more likely to be those with lower reward values, i.e., those with lower weighted available material, in which visiting them does not compensate for the effort to get there.

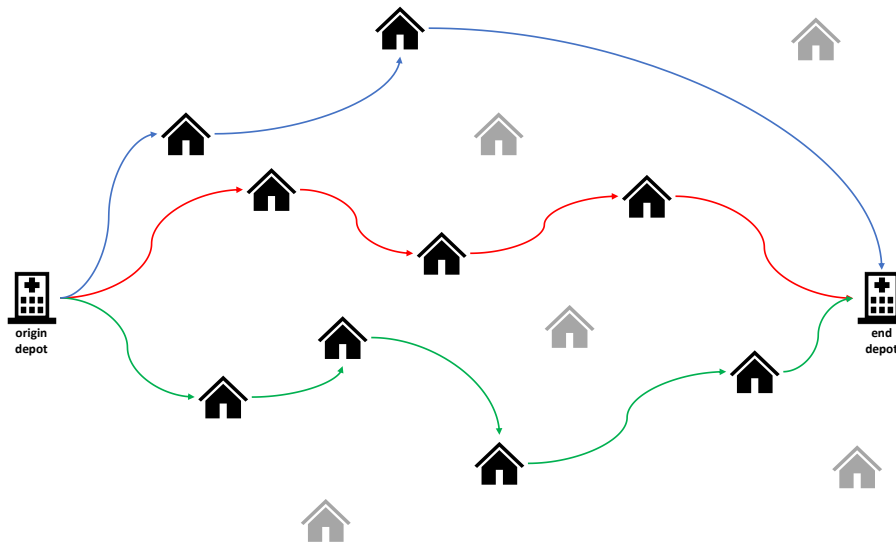


Figure 8.8: An example of a TOP solution with different start and end depots.

8.1.3.3 Rich Pick-up and Delivery Routing Problems

The general PDP combines the OVRP and the TOP. This problem was first addressed in the literature by Savelsbergh and Sol, 1995, and concerns with defining a set of optimal routes to satisfy a set of transportation requests –each requiring both pick-up and delivery under capacity and precedence constraints–, in order to minimize transportation costs. The transportation is performed from a set of origins to a set of destinations, without any transshipment at other locations. The origins represent the pick-up locations, while the destinations are the delivery points.

The classical PDP has been widely extended by incorporating several characteristics into the problem in order to bring it as closest as possible to real-life environments. Constraints such as time-windows to visit costumers (Aziez et al., 2020; Dumas et al., 1991), heterogeneous fleet of vehicles (Avci and Topaloglu, 2016), multi-depot (Sombuntham and Kachitvichayanukul, 2010), a combination of them (Bettinelli et al., 2014), and multi-echelon distribution (Martins et al., 2020c; Bayliss et al., 2020b), or even uncertainties (Györgyi and Kis, 2019) have been addressed into the classical problem, resulting in a large number of variants of this classical problem, frequently called as *rich* pick-up and delivery problems (RPDPs). According to Parragh et al., 2008, the term ‘rich’ refers to the additional features that are incorporated in order to accommodate the different characteristics of the various

problem types. By incorporating those features, the RPDP has become even harder to be solved to optimality, which results in the proposal of many approximated approaches for solving them. In our case, the problem addressed in this work can be described as a PDP by considering the last node to be visited (end depot) as the delivery node, while the remaining ones, except the first node (start depot), are pick-up points.

8.1.4 Biased-Randomized Algorithms

Heuristic and metaheuristic algorithms have become the default standard when dealing with rich and realistic vehicle routing problems (Fikar et al., 2016). Biased-randomized optimization algorithms make use of Monte Carlo simulation and skewed probability distributions to introduce a non-uniform random behavior into a constructive heuristic. Thus, the heuristic is transformed into a more powerful probabilistic algorithm, which can be run in virtually the same wall-clock time as the original heuristic if parallelization techniques are employed (Ferone et al., 2019). One of the main advantages of BR algorithms is their ability to generate multiple promising solutions that still follow the logic behind the original heuristic.

BR algorithms have been successfully used during the last years to solve different rich and realistic variants of vehicle routing problems (Dominguez et al., 2016c), permutation flow-shop problems (Gonzalez-Neira et al., 2017), location routing problems (Quintero-Araujo et al., 2017), facility location problems (Pages-Bernaus et al., 2019), waste collection problems (Gruler et al., 2017), arc routing problems (Gonzalez-Martin et al., 2012), horizontal cooperation problems (Quintero-Araujo et al., 2019), constrained portfolio optimization problems (Kizys et al., 2019), and e-marketing problems (Marmol et al., 2020).

A different class of BR algorithms was introduced by Gonçalves and Resende, 2011 for solving combinatorial optimization problems. Because its core is a genetic algorithm, the biased random-key genetic algorithms (BRKGA) aim to bias the selection of parents for generating new solutions. Recently, this solving methodology has been developed for solving an OVRP with capacity and distance constraints (Ruiz et al., 2019). In literature, the BRKGA has been also largely applied for solving different scheduling problems (Brandão et al., 2015; Brandão et al., 2017; Andrade et al., 2019; Homayouni et al., 2020).

8.1.5 Adapting Heavy Methods into Agile Algorithms

Metaheuristic frameworks are robust tools that provide optimal or pseudo-optimal solutions to optimization problems, which are typically large-scale and *NP-hard*, in a reasonable computing time. In order to obtain near-optimal solutions, these frameworks are composed of complex operators which are designed specifically to deal with the optimization problem to solve. Although these methods have been widely applied in the context of computational intelligence to cope with traditional optimization problems, they are not very useful when dealing with real-time and dynamic optimization problems that vary each day, like the ones faced in our hospital-logistics project. Hence, the concept of 'agile' optimization has arisen to deal with this new kind of problems, which are highly dynamic and require

flexible and even ‘light’ algorithms able to provide good solutions very fast without having to complete time-consuming set-up processes. This refers to a new optimization paradigm for *NP-hard* and large-scale optimization problems that need to be solved in ‘real-time.’ In this context, ‘agile’ algorithms follows the next principles: (i) extremely fast execution (i.e., seconds or even milliseconds); (ii) easy to implement and modify; (iii) flexible enough to deal with different problems and variants; (iv) parameter-less, avoiding complex and time-costly fine-tuning processes; and (v) specifically designed to run iteratively every few seconds or minutes, as new data is available.

In particular, agile optimization is based on the hybridization of biased-randomized algorithms and parallel computing. The intrinsic characteristics of biased-randomized algorithms make them a perfect candidate to be massively parallelized. In effect, they are good candidates to be executed using massively parallel processing architectures (Parhami, 2006). Usually, biased-randomized heuristics are wrapped into a multi-start framework, which is a sequential and iterative approach. Hence, a different solution is generated in each iteration. Typically, the multi-start methods are composed of two phases: a first one in which a new solution is generated using a deterministic constructive heuristic, and a second one in which the deterministic heuristic is turned into a probabilistic one, applying a skewed probability distribution to induce an oriented (biased) random behavior, without losing the logic behind of the original heuristic. Further, in this second step, the algorithm compares the newly generated solution with the best solution obtained so far –updating the latter whenever appropriate. The main idea behind agile optimization is as follows: instead of using a multi-start sequential approach, many threads of the biased-randomized algorithm can be run in parallel (e.g., using different computers or GPUs), as shown in Figure 8.9. As a result, several alternative solutions are generated in the same wall-clock time as the one employed by the original heuristic –that is, milliseconds in most cases. Then, different solutions are provided. Some of these solutions outperform the one generated by the original heuristic, while others show different characteristics that might be interesting for the decision-maker.

8.1.6 Solving Approach

Algorithm 24 shows our TOP-like algorithm used to solve the hospital-logistics problem. It works as follows: firstly, a dummy solution is generated (line 1), being this solution composed of one route per location (house). For each route, a vehicle departs from the origin depot, visits the location, and then returns to the destination depot. In case this route cannot be performed within the maximum driving time, its respective location is discarded from the problem since visiting this location is not possible. The second stage regards the computation of savings that are associated with each edge connecting two different locations (line 2). This computation considers both the travel time required to traverse that edge and the aggregated reward generated by visiting both locations. Since each edge is associated with two directions (arcs) in which it is traversed, individual savings are calculated for each arc. Then, these lists are sorted from higher to lower savings. Next, based on this sorted savings list, a route-merging process is started. The arc with the highest saving, i.e., that one at the top of the sorted list, is selected in each iteration (line 4). By using the selected arc, its two

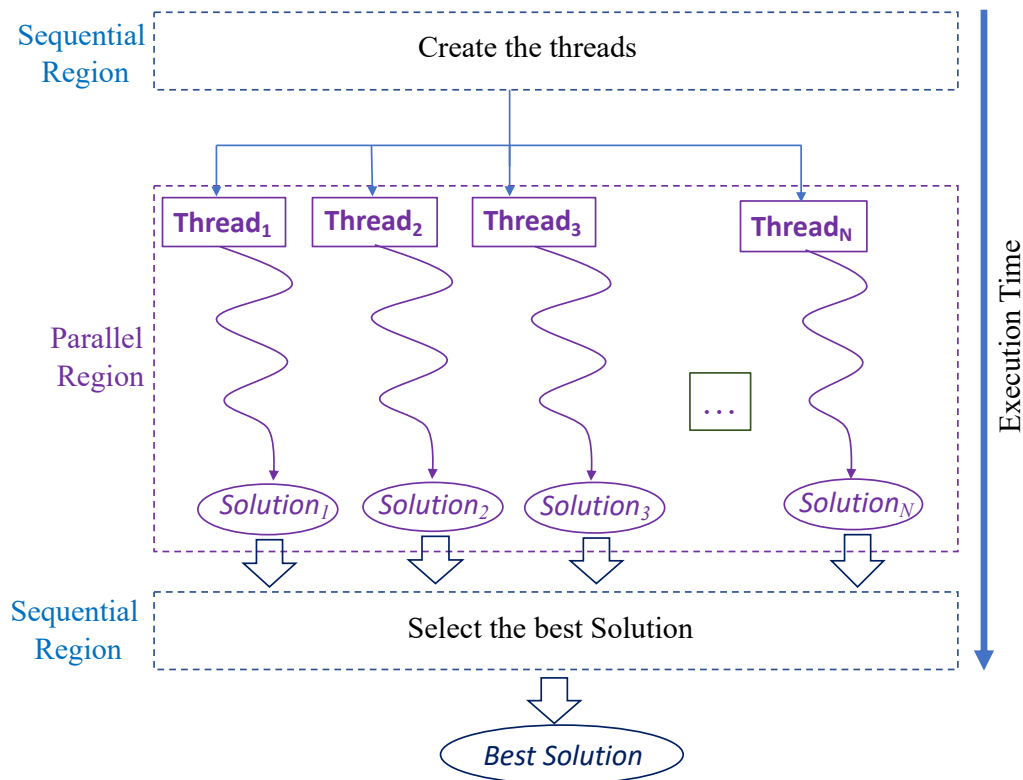


Figure 8.9: Parallel execution of a biased-randomized algorithm.

corresponding routes are merged into a new one, as far as this new route does not violate the driving-range constraint (line 9). The selected edge is later removed from the savings list, and this process is repeated until the list is empty. As a result, a list of routes is generated and sorted according to the total collected reward (line 15). Finally, from the sorted list of routes, the first n routes are selected, where n is the size of the vehicles' fleet. This heuristic is later extended into a probabilistic algorithm by introducing a biased-randomization behavior, which smooths the original greedy behavior of the heuristic. As explained in Section 8.1.4, biased-randomization techniques employ skewed probability distributions to induce an 'oriented' (non-uniform) random behavior into deterministic procedures, consequently transforming them into randomized algorithms while preserving the logic behind the original greedy heuristics. For doing so, we employ a geometric probability distribution with a single parameter β ($0 < \beta < 1$), which controls the relative level of greediness present in the randomized behavior of the algorithm. After a fine-tuning process, $\beta = 0.3$ has presented a good performance, being this value selected to be used in our computations.

8.1.7 Examples of Daily Routing Plans

Many variants of classical problems show increasing complexity due to the incorporation of extra decisions and operational constraints in order to reflect real-life cases. In our context, the problems are notably dynamic since they must be frequently modified, even on a daily basis. These daily challenges are determined by limited resources and a variety of operational decisions, such as:

Pseudocode 24: Example of one heuristic-based approach employed.

```

Data: set of nodes  $V$ 
1 Function LH( $V$ ):
2   sol  $\leftarrow$  generateDummySolution(Inputs)
3   savingList  $\leftarrow$  computeSortedSavingList(Inputs)
4   while (savingList is not empty) do
5     arc  $\leftarrow$  selectNextArc(savingList,  $\beta$ )
6     iRoute  $\leftarrow$  getStartingRoute(arc)
7     jRoute  $\leftarrow$  getClosingRoute(arc)
8     newRoute  $\leftarrow$  mergeRoutes(iRoute, jRoute)
9     timeNewRoute  $\leftarrow$  calcRouteTravelTime(newRoute)
10    isMergeValid  $\leftarrow$  validateMergeDrivingConsts(timeNewRoute, drivingRange)
11    if (isMergeValid) then
12      | sol  $\leftarrow$  updateSolution(newRoute, iRoute, jRoute, sol)
13    end
14    deleteEdgeFromSavingList(arc)
15  end
16  sortRoutesByProfit(sol)
17  deleteRoutesByProfit(sol, maxVehicles)
18  return sol
19 End

```

1. *A maximum tour length for drivers:* as they are volunteers (and also to avoid excessive exposition to risk), the total time that drivers can dedicate to pick up elements is limited. Routes must be as balanced as possible so that the work time of each driver is similar. Additionally, this total length must include a service time per visited node, which is assumed to be constant.
2. *A limited number of vehicles:* the number of volunteers is variable each day, which limits the number of routes that can be designed. This condition, jointly with the limit in the drivers' work time, makes that a few nodes must be skipped some days. These instances are then solved preferentially using a TOP-like algorithm.
3. *Origin and arrival nodes are the same or different:* most instances require an OVRP-like solution, in which drivers depart from a point that is different from the final destination. However, sometimes this constraint can be relaxed, as we will explain later.
4. *Mandatory nodes to visit:* the three previous constraints rely on a TOP-like algorithm, resulting in locations that are not visited. However, there are some cases in which specific nodes must be mandatorily visited. These mandatory nodes to visit are more likely to be the preparation nodes, the medical centers, or some makers who offer a large number of medical supplies.
5. *Segmentation of nodes:* drivers are more willing to visit the makers and medical centers depending on the geographical zone where they are located. Hence, a segmentation process is required, in which nodes are grouped in clusters. Besides, some instances include multiple origins or arrival depots, and each cluster must contain only one pair of origin-arrival. Whenever these conditions show, the solving process is semi-automatic in order to create properly the clusters.

6. *Precedence constraints*: sometimes it is mandatory to pass through a specific node to pick up supplies before making any delivery at the medical centers, e.g., to visit a preparation node before a hospital. Hence, this situation imposes mandatory precedence in a specific group of nodes.
7. *Pick-ups and deliveries*: despite most nodes are pick-up points, the loaded freight must be unloaded somewhere, either into an intermediate location during the routing operation, or, more commonly, at the end of this process. Therefore, routes are frequently characterized by both operations.

As we can notice, all this dynamism, in terms of daily-based problem conceptualization, enlarges the problems' complexity. Therefore, our solving methodologies must be flexible enough to deal with them smartly, quickly, and efficiently. Below, we re-describe the most common problem variants we faced during the development of this project, now based on their definitions and particularities (1)-(7), previously defined:

- **VRP**: As described, the consideration of this problem relies on operating scenarios in which the fleet of vehicles is large enough to visit all the problem locations without disrespecting any constraint regarding vehicle capacity and/or tour length. Moreover, the routes must necessarily start and end at the same point. By being the simplest problem, it is barely considered in its classical definition, resulting in frequent adaptations, which lead us to the following variants.
- **OVRP**: Unlike classical VRPs, which impose the routes starting and ending their trajectories at the same depot, OVRPs relax this constraint in order to allow different points for starting and concluding the operation of a route. It consists of another typical situation since the drivers usually start their routes from home –or any location– and finish them at hospitals or health care centers, where the collected loads must be delivered.
- **PDP**: Several requests must be processed daily, requiring both picking up and delivering of goods. This might be the case in which all cases rely on, where a set of pick-up locations are visited for loading and, later, this freight is unloaded at the delivery points, usually represented by the end depot.
- **TOP**: As described, a common application context that relies on the definition of TOP regards the situation where a limited number of vehicles are available to process a large number of visits. Therefore, as part of the decision to be made, a set of nodes must be discarded during the routing in order to satisfy the constraint related to the maximum tour length, which, implicitly, defines the number of needed vehicles.

Based on these VRP variants, Figure 8.10 presents a visual schema for classifying the studied problems according to their main particularities (1) to (7). Notice that the main core of each problem limits its definition. In other words, for instance, the OVRP and VRP are defined by use or not of constraint 3, respectively. The TOP is mainly defined by constraint

2, which might result in non-visited nodes. Finally, the PDP is defined by particularity 7. The remaining characteristics are considered, in some way, as minor, but no less important, part of the problem definition. By visualizing this classification schema, we can notice that these problems are commonly characterized by not only distinct singularities but also, in some cases, common components. Such particularities define which problem is the most suitable one for representing the realistic scenario.

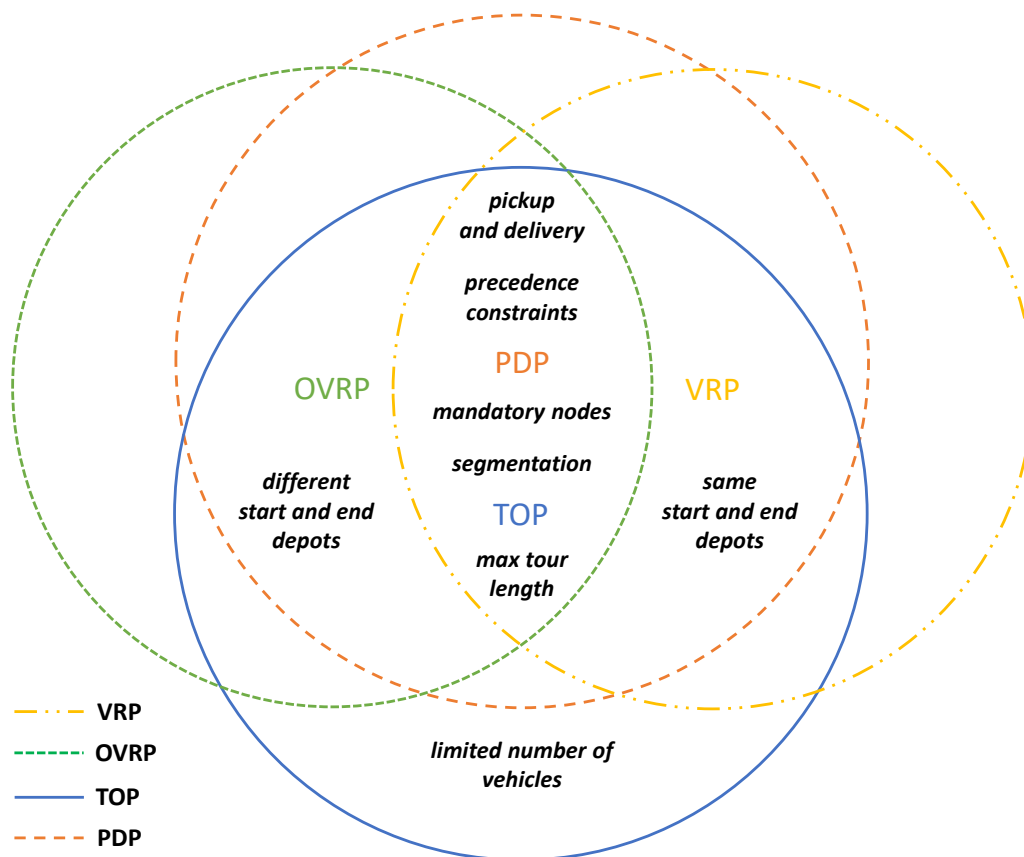


Figure 8.10: The visual classification of the problems regarding their specifications.

Taking into account that there was not service during most weekends –especially as the urgency for the new material was lower after the first weeks–, a total of 29 instances (days) were solved during this COVID-19 crisis. Table 8.1 displays both the known characteristics of each instance (input columns) and obtained results (output columns). Additionally, the instance name is shown, which corresponds to the date for which the instance was solved. Notice that the service time decreases to 4 minutes from the instance *apr-04*. Initially, coordinators estimated a constant service time per node of 7 minutes, however, drivers suggested a shorter time given the experience acquired in previous days. In general, the origin node is not the same as the arrival node. Nevertheless, some instances allow relaxing this constraint. They are marked with an asterisk in Table 8.1. Strictly speaking, these instances

correspond to an OVRP. However, since the arrival node must also be visited before starting the route, i.e., there are *mandatory nodes* and *precedence constraints*, it was possible to adjust these instances in the pre-processing step so that a VRP-like problem was solved.

The output columns in Table 8.1 show the maximum tour length (MTL) and the number of visited nodes according to the results obtained by each algorithm. As the VRP algorithm is designed to visit always all points, the number of nodes in the corresponding column represents the total input nodes in the instance. Conversely, the number yielded by the TOP algorithm is less than or equal to the total nodes. Skipping nodes is necessary for some instances given the limitations in both tour lengths and available vehicles. For example, the instance *apr-13b* imposes that the single available vehicle must not take more than 6 hours in completing its tour. The VRP algorithm yields a total travel time of 6 hours and 34 minutes to visit 24 nodes, which violates such constraint. Alternatively, the TOP algorithm designs a 21-node route that takes 5 hours and 54 minutes. Therefore, the TOP algorithm is the best strategy to solve this instance.

The hardness of the travel time constraint depends on the problem instance since drivers are not the same every day. Hence, the travel time is a soft constraint in instances *apr-04* and *apr-17*. Anyway, the TOP algorithm is the best strategy in these 2 instances since the time yielded by the VRP algorithm is prohibitively high. The rest of the instances have the total travel time as a hard constraint. For most of them, the VRP algorithm is the best strategy, because it yields a shorter time than the TOP algorithm, guaranteeing complete routes visiting all nodes. Figure 8.11 shows an example of the best routes obtained by VRP and TOP algorithms for the instance *apr-08*. Three nodes are skipped in the second case to meet the time constraint of 5 hours, which generates savings of 35 minutes with respect to the first case. This example shows the advantages of using a TOP algorithm when the time is limited since it finds a good balance between the reward offered by each node and the cost of service.

8.1.8 Managerial Insights

This experience shows how synergies appear when there is a diverse team working together towards a challenging and common goal. Firstly, through the combination of the know-how from research and scientific field and the one from business and entrepreneurship contexts. Secondly, through the collaboration between companies (logistics), institutions (the ICSO@IN3 research group), and volunteers (the “coronavirus makers” community), with very different organizational conditions that provide a broader scope of the project benefits. For example, the fast experiential feedback from the field creates a rich feedback loop in order to tailor further the process followed at all levels.

We have also seen how a new restriction appearing in the real-use cases may need a completely different approach to its solution. Such approaches have been evolved through years of research. However, once the knowledge exists, even in a more theoretical context, it can be quickly applied to the real use case by means of the know-how and experience from the researchers and a clear explanation of the need from the field side.

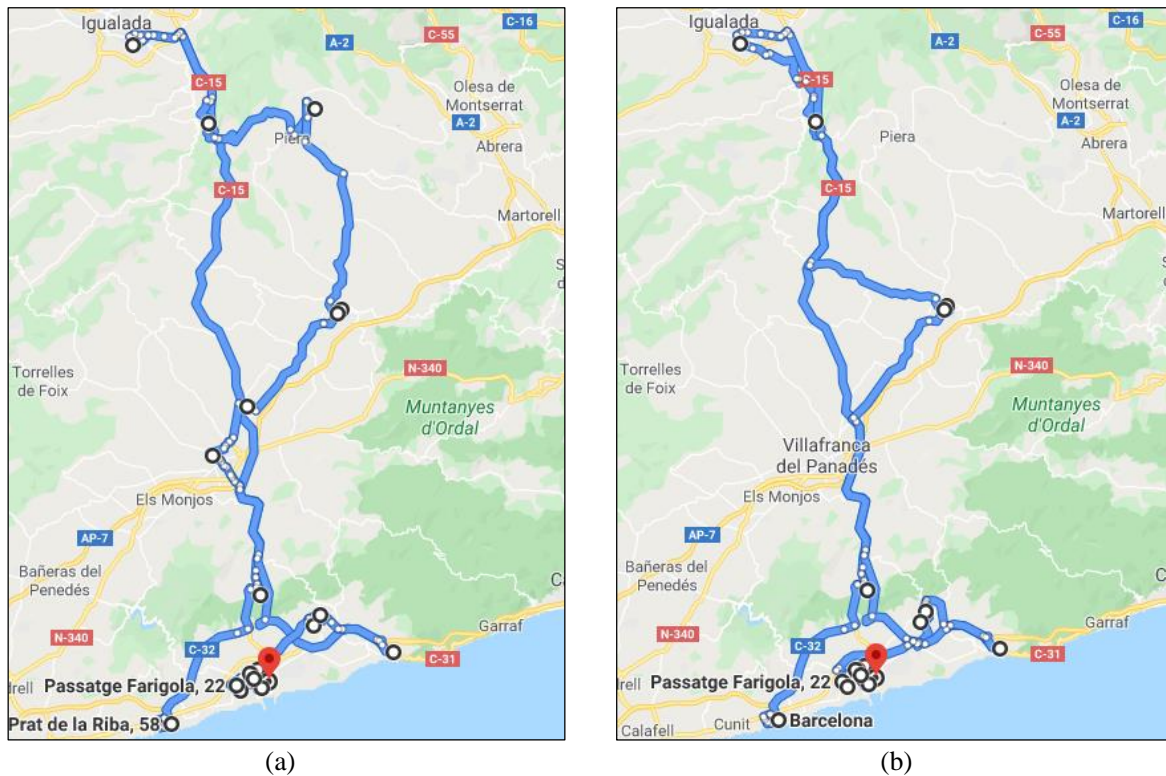


Figure 8.11: Routes generated for the instance *apr-08* by VRP(a) and TOP(b) algorithms.

8.1.9 Conclusions

This chapter has discussed how computational intelligence has been useful support for dealing with complex logistics challenges during the COVID-19 pandemic. In particular, we describe a case study regarding the metropolitan area of Barcelona (Spain) during March, April, and May 2020, where the pandemic was at its high and hospitals did not have enough sanitary material to protect their nurses, doctors, and other staff. Under those critical circumstances, a self-organized community of “makers” was able to use 3D printers at their homes to generate thousands of face covers, open-door devices, and similar sanitary items. The challenge of collecting these items from hundreds of individual houses and using a limited fleet of vehicles and a threshold time per service was huge, not only due to the size of the collection-routing problem but also mainly to the fact that the problem was evolving day after day. Thus, while some days the problem was more similar to a rich vehicle routing one, other days it needed to be modeled as a rich team orienteering problem.

In order to cope with this optimization challenge, which consisted of a different problem every day that needed to be solved in a few minutes, our team of researchers adapted some of the vehicle routing and team orienteering metaheuristic algorithms. The adaptation consisted in transforming ‘heavy’ algorithms into flexible and agile ones capable to adapt themselves –with little or no extra effort on our side– to the new characteristics of the problem, which were changing every day. The experience has shown us that: (i) in crisis scenarios like the one described in the chapter, a more ‘agile’ optimization paradigm

is requested –in contrast to the use of complex algorithms that focus on the solving of a single optimization problem, more flexible and fast algorithms are needed; and (ii) rapid development environments, such as those provided by the Python programming language, might significantly reduce the effort and time required to adapt algorithms to new variants of routing optimization problems.

Table 8.1: Instances' inputs and outputs.

Instance	Input						Output						Best strategy
	Number of clusters	Maximum tour length (h)	Service time (min)	Number of vehicles	Mandatory nodes	Precedence constraints	Same origin-arrival node	VRP algorithm		TOP algorithm		Best strategy	
								MTL(hh:mm)	Visited nodes	MTL(hh:mm)	Visited nodes		
mar-25	1	5	7	6				3:50	95	3:52	95	VRP	
mar-26	1	5	7	4				4:38	77	4:59	77	VRP	
mar-27	2	5	7	6				3:36	62	3:43	62	VRP	
mar-28	1	5	7	2				4:59	48	5:09	48	VRP	
mar-30	1	5	7	2	•			4:01	32	4:15	32	VRP	
mar-31	2	7	7	2	•			6:32	53	6:36	53	VRP	
apr-01	2	7	7	2		•		6:39	31	6:41	31	VRP	
apr-02	2	5	7	2				4:58	29	5:02	29	VRP	
apr-03	1	5	7	2				4:05	29	4:14	29	VRP	
apr-04	1	4	4	1	•			5:56	22	4:07	14	TOP	
apr-06a	2	6	4	2				5:58	46	6:07	46	VRP	
apr-06b	1	5	4	1	•			3:12	15	3:17	15	VRP	
apr-07	1	5	4	1	•			4:35	17	4:36	17	VRP	
apr-08	1	5	4	1	•			5:28	19	4:53	16	TOP	
apr-09a	2	6	4	2				6:15	51	5:54	49	TOP	
apr-09b	1	6	4	1	•	•		5:43	22	5:48	22	VRP	
apr-10	1	6	4	1	•	•		5:21	25	5:31	25	VRP	
apr-13a	2	5	4	2				4:59	39	5:05	39	VRP	
apr-13b	1	6	4	1				6:34	24	5:54	21	TOP	
apr-14	1	6	4	1	•		•	5:30	21	5:33	21	VRP	
apr-15	1	7	4	1	•	•		8:38	39	6:32	28	TOP	
apr-17	1	6	4	1	•	•		6:53	33	6:22	29	TOP	
apr-18	2	5	4	2				4:55	38	4:54	38	TOP	
apr-20	1	5	4	1	•	•		5:32	21	4:48	17	TOP	
apr-22	1	5	4	1	•	•		6:33	21	4:46	16	TOP	
apr-25	1	5	4	1	•	•		5:37	23	4:59	20	TOP	
apr-28	1	5	4	1	•	•		6:06	23	4:54	19	TOP	
apr-30	1	6	4	1	•	•		5:34	14	5:40	14	VRP	
may-02	1	5	4	1				3:30	13	3:36	13	VRP	

Chapter 9

Car-Sharing Systems

This chapter ¹ studies Ride-Sharing operations in smart cities. Unlike previous chapters that focus on the application of optimization solution methods to a range of COPs, this chapter provides a survey on ride-sharing optimization problems. Challenges in related problems are discussed, encouraging the use of agile algorithms to cope with these dynamic systems.

9.1 Introduction

Transport and logistics activities represent a key sector in modern societies, and they significantly contribute to their social and economic progress. At the same time, the raise of the on-demand economy (services) and the e-commerce activity (products) has boosted the number of pick-ups and deliveries in urban, metropolitan, and peri-urban areas. Thus, there is a need for increasing the effectiveness and sustainability of T&L activities and policies (Cui et al., 2020). Due to the increasing number of people who live in urban areas, many local and regional governments realize that T&L activities will play a major role in the development of the so-called smart sustainable cities (Bibri and Krogstie, 2019). Large quantities of data are gathered in real-time via electronic devices located inside vehicles and infrastructures (computer chips, sensors, traffic cameras, drones, etc.), transmitted over the Internet, and analyzed through information and expert systems (Mehmood et al., 2017). Monetary, environmental, and social costs associated with single occupancy vehicles could be reduced by more efficient utilization of empty seats in personal transportation vehicles. This is the goal of carpooling and ride-sharing strategies, which, apart from generating substantial economic impact to users, aim at reducing the number of vehicles on the road and, as a consequence, contribute to diminishing traffic and pollution (Bistaffa et al., 2019). According to Schrank et al. (2019), the annual cost of congestion in the United States (U.S.) achieved \$ 166 billion in 2017, which caused Americans to lose around 8.8 billion hours on sitting in traffic and purchase an extra 3.3 billion gallons of fuel. Environmentally speaking, transportation counted for about 28% of total CO₂e in the U.S. in 2018, being light-duty vehicles responsible for 59% of them (United States Environmental Protection Agency, 2020). In

¹The contents of this chapter are based on the following work:

- **Martins, L. C.**; de la Torre, R.; Corlu, C. G.; Juan, A. A.; Masmoudi, M. A. (2021): [Optimizing ride-sharing operations in smart sustainable cities: Challenges and the need for agile algorithms](#). *Computers & Industrial Engineering*, 153, 107080.

Europe, on the other hand, transportation was responsible for almost 30% of CO_2e in 2016, of which 72% comes from road transportation. Particularly, cars are responsible for almost 61% of these 72% of gas emissions (European Environment Agency, 2019). In an effort to minimize related problems, such as greenhouse effects and global warming, the European Union developed a strategic plan for low-emission mobility. As stated in European Commission (2016), one of the main elements on which this strategy relies on refers to increasing the efficiency of the transport system by benefiting from digital technologies, smart pricing, and further encouraging the shift to lower emission sustainable transportation modes. Therefore, the need for smarter and sustainable transportation modes is clear, whose development has been possible thanks to recent advances in communication and information technologies.

Carpooling and ride-sharing are two of the main peer-to-peer (P2P) services in car-sharing. P2P services followed the diffusion of smart-phone technology and social networking websites (Prieto et al., 2017), transforming car-sharing services into an international transportation trend. Such services rely on sharing privately owned vehicles for a particular trip in the surrounding area on an hourly or daily basis (Ballús-Armet et al., 2014).

The seminal studies regarding the use of ride-sharing systems are dated to 70s. According to Kornhauser et al. (1977), the first motivation for adopting a ride-sharing system was the fuel crisis of 1973 in the U.S., and the scarcity of federal funds for implementing new urban transport facilities. At the time, the increase of vehicles' utilization in private transport represented the most obvious target for improving the systems' efficiency without constructing new physical facilities. Since the work of Kornhauser et al. (1977), the use of ride-sharing systems has been studied and gained considerable attention from researchers. Different types of ride-sharing can be identified in the literature: (i) ride-sharing with static requests –in which all requests are known before the trip starts (Yu et al., 2019); (ii) ride-sharing with dynamic requests –where new requests can be added during the execution of the transport service (Simonetto et al., 2019); and (iii) ride-sharing with either deterministic or stochastic requests (Long et al., 2018). Researchers around the world have studied many variants and real-life applications in cities such as New York (Schaller, 2017), Atlanta (Agatz et al., 2011), Rome (Naoum-Sawaya et al., 2015), Beijing (Ma et al., 2013), or Tokyo (Do et al., 2016). Several optimization techniques have been used to solve ride-sharing problems, including exact and approximate methods, as well as agent-based and dynamic simulation. Also, surveys on the different variants and applications of ride-sharing problems can be found in Furuhata et al. (2013) and, more recently, in Mourad et al. (2019). Because real-world problems are often dynamic and large-scale, car-sharing related problems are challenging. According to Borcuch (2016), a challenging task in developing car-sharing systems in real-world is the scaling of the shared-transportation problem-solving approach, in order to solve large-scale problems, like those required in real-life, where over thousands or millions of requests should be assigned.

Considering the aforementioned, the main contributions of this chapter can be summarized as follows: (i) we provide a review of recent works on optimization problems related to ride-sharing and carpooling, classifying them according to the employed solving

methodology (i.e., either exact, approximate or simulation methods); (ii) from the previous review, the main challenges are identified, specially in the context of smart and sustainable cities –including the increasing trend in considering self-driving and electric vehicles; and (iii) the concept of ‘agile’ optimization is discussed. Agile optimization algorithms are able to provide high-quality solutions in real-time by combining biased-randomized algorithms (Grasas et al., 2017) with parallel computing (Malapert et al., 2016). By taking advantage of these two approaches, the resulting methodology is capable of efficiently responding to every piece of new information that is being continuously incorporated into the system.

9.2 Car Sharing Activities

With the rise of mobile technology, car-sharing services have become an international transportation trend that holds the capability of significantly reducing congestion on the roads, diminishing traffic and pollution, and other externalities caused by the individual transportation. At the same time, these P2P systems allow users to accomplish several transportation goals, which include economic (e.g., costs reduction) and convenience interests (e.g., flexibility and speed), by allowing drivers and riders to share the associated costs (e.g., fuel, tolls, parking fees) so that each benefits from the shared ride (Stiglic et al., 2015). Although both systems allow users to travel together and share transportation costs, carpooling often limits users to consistent schedules. It also fixes riders to the same place at the same time. Consequently, the full potential of prearranged carpooling is often constrained by these operational limitations (Kornhauser et al., 1977). Ride-sharing, on the other hand, allows for more flexible schedules and locations. In both carpooling and ride-sharing services, users share rides provided by drivers, who are participating individuals that operate with their private vehicles. Both services charge passengers with a fee to share the ride. The following two sections discuss ride-sharing and carpooling services in more detail.

9.2.1 Ride-sharing

By being an automated process in which a service provider matches travelers with similar itineraries and time schedules to share a ride on short-notice in a personal vehicle, ride-sharing systems are naturally dynamic (Prieto et al., 2017). Their complexity relies mainly on matching individuals subject to spatiotemporal constraints, which must be specified from both parties –i.e., drivers and users– before the desired ride is established and executed. On the one hand, passengers request a ride at a specific time, from a specific origin to a specific destination. On the other hand, drivers have a fixed trajectory and departure time. Consequently, ride-sharing systems require certain sort of flexibility since deviations might be needed at different points on the trajectory in order to pick up and drop off passengers, as long as these detour distances do not exceed the driver’s distance tolerance (Cici et al., 2015).

Figure 9.1 presents an illustration of a ride-sharing trip. In Figure 9.1a, solid lines represent the fixed trajectory of the driver, in which passenger 1 walks to driver’s origin, and

passengers 2 and 3 are picked up during the driver's trajectory to destination. Figure 9.1b represents an extension of the previous one, where dashed lines represent possible detour deviations that might occur, for instance, in order to: (i) pick up a passenger (the passenger a); and/or (ii) drop off a boarded passenger (the passenger 1) at a location which is different from the driver destination (location b). These two detours are done by considering a maximum threshold of locomotion. From the passengers' point of view, this process implies a wait until the driver's latest departure time.

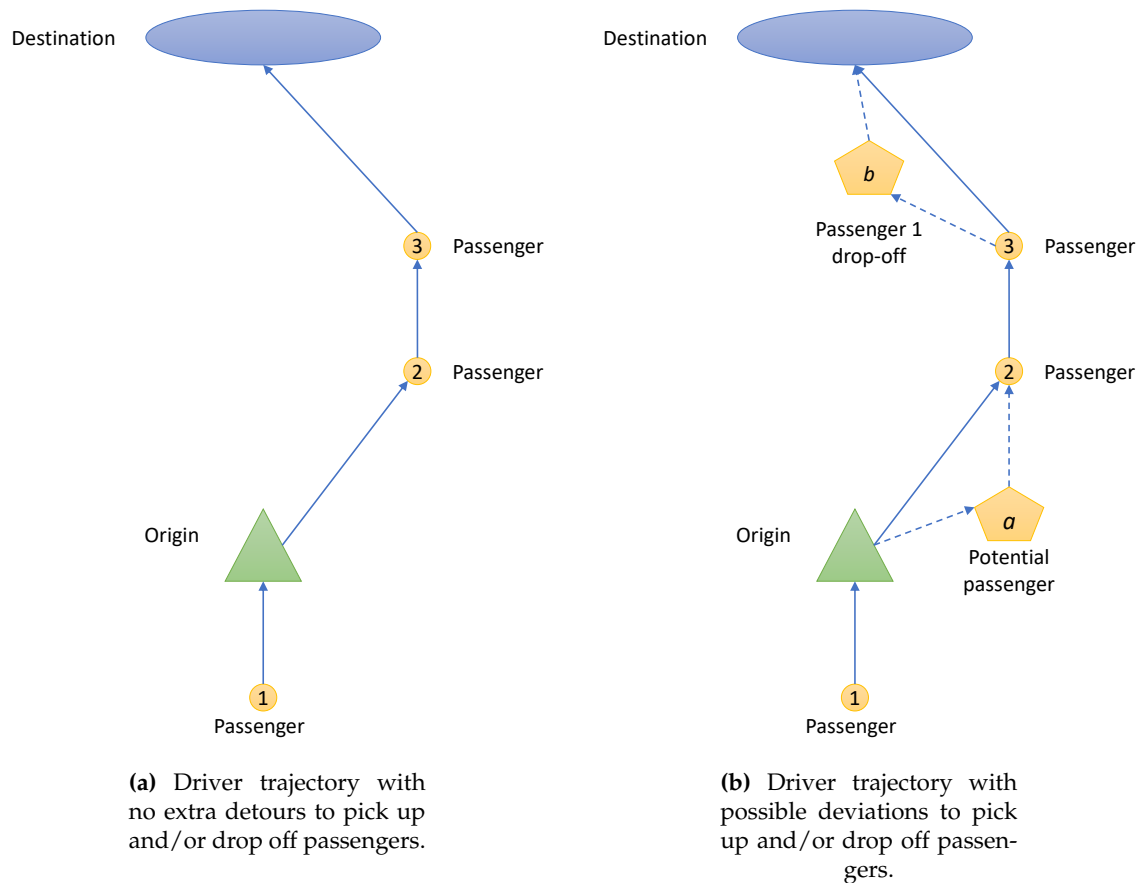


Figure 9.1: A visual representation of two possible ride-sharing rides.

9.2.2 Carpooling

Unlike ride-sharing activities, carpooling rides are less flexible activities that aim to transport simultaneously several people from a common starting point to a common end point (Nechita et al., 2016), with the main goal of saving money. These services encourage commuters who are moving in the same direction to share private vehicles (Duan et al., 2018). According to Nechita et al. (2016), the most usual situation of carpooling occurs when neighbors work at the same facility and agree to travel using only one car in order to share the travel expenses. Two variants of such carpooling exist: (a) the pools sharing a ride to work

also share the same ride returning from work back home, and (b) both *to-work* and *return-from-work* are treated as different problems, and, hence, must be solved independently (Baldacci et al., 2004).

In carpooling systems, by fixing both the origin and destination locations that define the trajectories, riders are fixed to be at the same place at the same time before starting the ride. Therefore, users are limited to consistent schedules. In these systems, the origin and destination are announced by the drivers, and no deviations, pick-ups, or drop-offs are allowed during the execution of a ride. According to Stiglic et al. (2015), the use of meeting points in car-sharing, such as those from carpooling activities, increases the feasible matches between drivers and riders, apart from allowing the driver to be matched with multiple riders without increasing the number of stops the driver needs to make. In Figure 9.2, a carpooling ride is presented, where three passengers move to the origin, where the trip is started (i.e., the start point), and they get to the destination together with the driver.

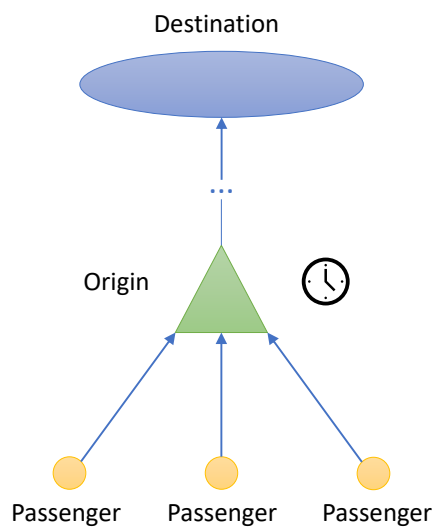


Figure 9.2: A visual representation of a possible carpooling ride.

9.3 Research Questions & Initial Classification

In the context of ride-sharing activities inside smart sustainable cities, this section describes the methodology adopted in carrying out a systematic literature review (Tranfield et al., 2003). Following Thornhill et al. (2009), our review is conducted using an iterative method, which consists of the definition of appropriate keywords, the search within the current literature, and its analysis. This systematic approach helps to reduce any bias and ensures the reproducibility of the process (Cook et al., 1995).

The first step includes defining the main research goals and objectives, selecting the database and the keywords, and designing the search process. We aim to answer the following research questions: (i) which are the main optimization challenges related to ride-sharing activities in smart and sustainable cities? and (ii) what optimization and simulation

techniques have been employed so far in ride-sharing and carpooling optimization problems? The following keywords were proposed: ride-sharing, smart sustainable cities, carpooling, and optimization problems. We have analyzed publications included in journals indexed within the Science Citation Index (SCI) and the Social Science Citation Index (SSCI), both part of the Web of Science (WoS), which is considered to be one of the main sources of information in the academia (Newbert, 2007). Through the analysis of the literature presented in the Sections 9.4, 9.5, and 9.6, answers will be depicted to these questions. According to Rodríguez Bolívar et al. (2010), books (including reviews), editorials, brief communications, letters to the editor, symposiums, and articles of a professional nature, provide a limited view of the subject, and therefore, must be excluded from the analysis. However, our review does take into consideration articles published in special issues of journals, since those actually reflect a great interest in the study of any issue.

A set of inclusion/exclusion criteria have also been implemented. The first one makes reference to the inclusion of those papers related to car-sharing, ride-sharing and carpooling in smart cities. The second one deals with the consideration of works that actually apply optimization techniques and/or metaheuristics. The last criterion was based on the inclusion of those articles that explore agile optimization to solve transportation problems. From the results of the first step, a total of 1,355 papers were found. In fact, the analysis of citations of the original elements in the Web of Science collection related to “ride-sharing”, “car-sharing” and “carpooling” terms, shows a growing interest (see Figure 9.3). Similarly, Figure 9.4 shows, for the terms ‘ride-sharing’, ‘carpooling’, and ‘car-sharing’, the time evolution of the number of articles indexed in the WoS.

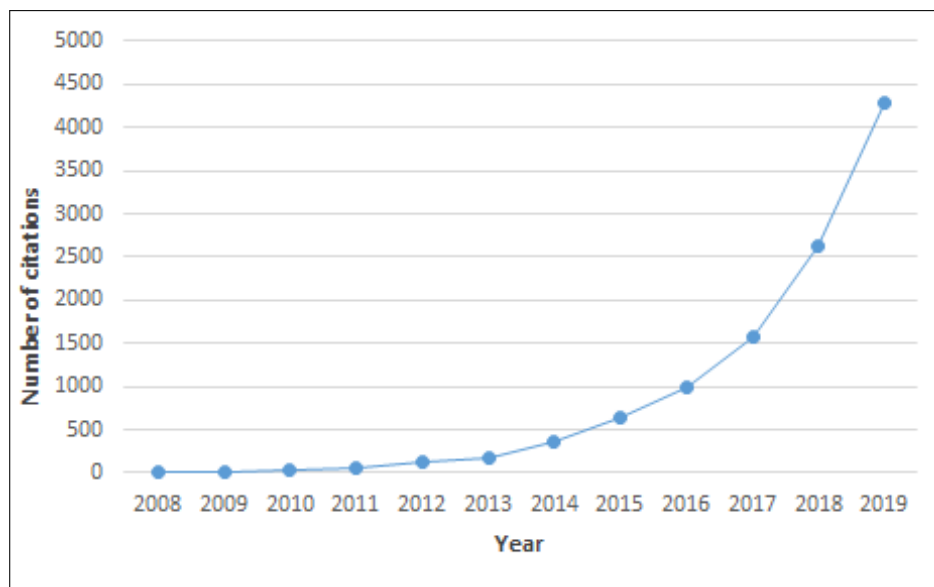


Figure 9.3: Number of citations per year indexed in WoS.

The second step consists of classifying the references gathered from the performed search (Hartley and Kostoff, 2003). All in all, a total of 86 papers were selected to be analyzed. Regarding the journals in which the selected papers were published, the research reveals that the most common journals comprise a total of 32.94% (29 papers): *Transportation Research*

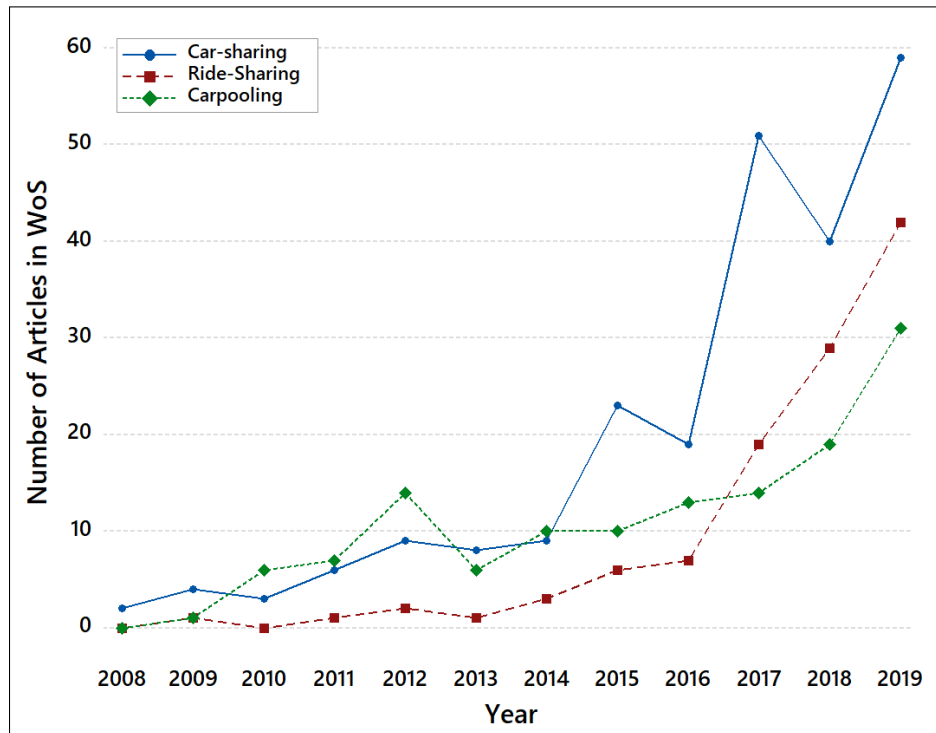


Figure 9.4: Number of scientific articles per year indexed in WoS.

Part B: Methodological (13 papers); Transportation Research Part C: Emerging Technologies (6 papers); Computers & Industrial Engineering (5 papers); and European Journal of Operational Research (4 papers). The dominant research areas are directly connected with the topics assessed in Section 9.2: Transportation (53.2%), Engineering (46%), Computer Science (43.2%), Business Economics (22.2%), Mathematics (15.6%), and Environmental Sciences Ecology (13.4%).

In the third and last step, the selected works were classified according to the analytical method used, i.e., exact, approximate or simulation approaches. Also, these works were examined to identify the addressed main challenges.

From an analytical perspective, most of the works on ride-sharing and carpooling optimization are identified as *NP-hard* problems. These problems are usually formulated as MILP/MIP models, and small-sized instances are solved using exact methods. However, due to the complexity of the ride-sharing problems, approximate methodologies have been used in the literature to solve large-sized instances as well. The next sections provide a review based on the solving approach employed. Another important aspect identified in the literature refers to whether the ride-sharing system is static or dynamic. In dynamic ride-sharing systems, trip information –which includes users’ origin, destination, and time schedule– is sent to the platform. Then, the solving methodology must match up drivers and riders on a very short notice, or even *en-route* (Agatz et al., 2012). Many studies are focused on developing algorithms for dynamic ride-sharing systems. Table 9.1 presents a classification of the reviewed articles by problem variant (static or dynamic) and solving methodology (exact or approximate). The following sections discuss in detail each of these works. In each section, we first introduce the related literature in the field of ride-sharing

problems, followed by the related literature in carpooling problems.

Table 9.1: Classification of ride-sharing articles by version (static or dynamic) and solution methodology.

<i>Study</i>	<i>Car-Sharing Problem</i>		<i>Methodology</i>		
	Static	Dynamic	Exact	Heuristic or Metaheuristic	Other
Kornhauser et al. (1977)		•			•
Baldacci et al. (2004)	•		•	•	
Agatz et al. (2011)		•	•		
Yan and Chen (2011)	•			•	•
Herbawi and Weber (2012)		•		•	
Hosni et al. (2014)	•	•	•	•	
He et al. (2014)	•				•
Lee and Savelsbergh (2015)		•	•	•	
Fagnant and Kockelman (2015)		•		•	•
Santos and Xavier (2015)	•	•		•	
Huang et al. (2015)	•		•	•	
Naoum-Sawaya et al. (2015)	•		•		
Stiglic et al. (2015)	•		•		
Schreieck et al. (2016)		•			•
Jung et al. (2016)		•		•	
Alonso-Mora et al. (2017)		•	•	•	
Levin et al. (2017)	•	•		•	
Masoud and Jayakrishnan (2017)	•		•		
Najmi et al. (2017)	•	•		•	
Wang et al. (2017a)		•	•	•	
Li et al. (2018b)	•		•	•	
Long et al. (2018)	•				•
Ma et al. (2018)	•			•	
Lokhandwala and Cai (2018)	•			•	•
Yu et al. (2019)	•		•		
Chen et al. (2019)	•		•	•	
Simonetto et al. (2019)		•		•	
Li and Chung (2020)	•		•	•	
Cheikh-Graiet et al. (2020)		•		•	

9.4 Exact Methods for Car Sharing Optimization

Despite their limitations for solving large-sized *NP-hard* optimization problems in short computing times, the use of exact methods is still relevant since they can be utilized to validate approximate methods in small-sized instances, as well as to provide lower and upper bounds to optimal solutions.

9.4.1 Ride-sharing

Ride-sharing systems are naturally dynamic because they require matching of travelers with similar itineraries and time schedules on short-notice (Prieto et al., 2017). Consequently, exact approaches for solving this class of problems must be flexible enough to deal with such particularities, since most of them require solutions in real-time. Considering a dynamic single-trip ride-sharing problem, Agatz et al. (2011) proposed an optimization procedure to match up drivers and riders on a very short notice and to determine the best set of proposed

ride-share matches. In the addressed environment, new drivers and riders continuously enter and leave the system. The objective is to minimize the total-wide vehicles-miles and the total travel cost, i.e., to maximize the total revenues of the provider. The proposed methodology is based on a rolling-horizon strategy, which incorporates an optimization procedure to determine the best set of ride-sharing matches by using the CPLEX commercial optimization software. To test their methodology, the authors performed a simulation based on travel demand data for the Atlanta metropolitan area. For cases in which the instances cannot be solved to optimality quickly, the authors defined a maximum solution time limit or an optimality gap which guarantees the finding of high-quality solutions. As expected, the simulation results improved basic greedy matching rules and suggested that the use of dynamic ride-sharing systems is able to reduce the overall travel cost of the system, as well as travel times of passengers. Likewise, Hosni et al. (2014) proposed a Lagrangian decomposition approach to maximize the total profit in a ride-sharing problem –i.e., their goal was to minimize the vacant seats, taxi fares to passengers, and number of vehicles. In this problem, customers request rides from specific pick-up locations to specific drop-off locations. Therefore, the optimal assignment of passengers to taxis must be determined, as well as the optimal route for each taxi. The Lagrangian approach decomposes the problem into sub-problems that are independently solved. Recently, Li and Chung (2020) introduced a novel deterministic model for the ride-sharing under travel time uncertainty, which addresses different origins and destinations of drivers and riders. Similar to the previous study, the objective aims to find optimal matches between riders and drivers, besides finding the optimal routes for drivers in which multiple drivers and multiple riders are considered. The model was solved through a MIP model using the *Gurobi* solver (Gurobi Optimization, LLC, 2020). Apart from being able to find optimal solutions, several hours to several days were needed to solve problems of up to 44 nodes. In order to overcome this shortcoming, some authors have proposed a hybrid method. Similarly, Naoum-Sawaya et al. (2015) studied a stochastic ride-sharing scenario by considering the unforeseen event of the car unavailability. They proposed an exact integer programming (IP) model to solve the problem. Lee and Savelsbergh (2015) focused on understanding the required budget to achieve a certain service level, in terms of serving a minimum percentage of riders, in a dynamic ride-sharing system. This budget is related to the cost of employing dedicated drivers, who are only addressed when the number of passengers increases to a certain level. Similar to previous authors, they formulated the problem as an IP model, which was solved by commercial IP solvers in order to validate a proposed metaheuristic approach. Li et al. (2018b) also proposed a MILP model to solve a ride-sharing problem, resolved by the CPLEX, and to validate a metaheuristic approach. Another example of dynamic ride-sharing can be found in Wang et al. (2017a), in which the passenger has the option of accepting or declining the assigned vehicle. They also proposed different mathematical programming approaches to find the best stable solution.

In the context of autonomous vehicles, Alonso-Mora et al. (2017) presented a mathematical model for solving, in real-time, high-capacity ride-sharing problems via dynamic trip-to-vehicle assignments. The authors proposed a reactive-anytime-optimal algorithm. Based on a greedy assignment, this algorithm returns a valid assignment of travel requests

to vehicles. Then, refines it over time, converging to an optimal solution. The optimal routes are dynamically generated with respect to online demands and vehicle locations by solving an ILP model. Later stages aim to balance the remaining idle vehicles, which are in areas far away from the one with an active request. The authors concluded that the using the ride-sharing concept can provide a substantial improvement in urban transportation systems by reducing the fleet size substantially. They also conclude that system parameters –such as vehicle capacity and fleet size– have a direct influence on the service quality and demand. Masoud and Jayakrishnan (2017) proposed an exact method, based on a decomposition algorithm, to solve a multi-hop ride-sharing problem. The objective is to minimize the total traveling cost, which also includes the fixed cost of the vehicles and the penalty cost of the non-served passengers. Chen et al. (2019) proposed an ILP formulation to solve the ride-sharing problem considering return restrictions to satisfy the business needs, meeting points, and the option for riders to transfer between drivers. However, the efficiency of this method was limited by the size of the instances, which were optimally solved just for cases with up to 80 participants –the computational time was set to a maximum of two hours. Finally, Yu et al. (2019) investigated a green ride-sharing problem whose multi-objective function consists in maximizing the average ride profit of the drivers and minimizing the carbon emissions. An exact method –based on cutting the non-Pareto-optimal solutions using a decomposition approach– was developed to solve this problem. The method was tested on benchmark instances for the pick-up and delivery problem with time windows, which were initially proposed by Li and Lim (2003).

9.4.2 Carpooling

Baldacci et al. (2004) proposed both an exact and heuristic method for solving a single way, referred to as a *to-work*, carpooling problem. The exact approach is based on a bounding procedure that combines three lower bounds derived from different relaxations of the problem. Two different problem formulations were presented. The first one is based on three-index decision variables specifying the arcs traversed by each car while the second one uses a set-partitioning formulation whose variables correspond to feasible paths for the cars. The paths were generated by dynamic programming, and for solving the formulations, the CPLEX solver was employed as the integer programming solver in the exact method. The authors tested the approaches to VRP derived instances, where the majority of the problems could be solved by the exact approach in reasonable computing times. Moreover, the proposed bounding procedure showed to be competitive with other column generation methods. Years later, Stiglic et al. (2015) introduced an IP formulation to formulate a single driver, multiple riders ride-share matching problem, in order to maximize the number of matched participants in large-scale ride-sharing systems with meeting points. The authors designed and implemented an algorithm that optimally matches drivers and riders, in which each driver has the possibility of having at most only one place to pick-up passengers, and one place to drop-off them, which characterizes a carpooling service. In other words, passengers will be taken at the same time in a common node, and they will also be dropped off at the same time in another common node. The pick-up and drop-off

nodes are set at strategic locations or transit points –such as bus stops, refueling stations, etc. This allows passengers to be picked up by other drivers on their way to their final destination. The CPLEX was shown to solve the integer programs –i.e., the matching process– in a few seconds in all tested settings, which suggests that the algorithm is appropriate for use in practice. This concept has been also studied by other researchers, such as Li et al. (2018b), Stiglic et al. (2018), and Khademi Zareh et al. (2019).

9.5 Metaheuristic Methods for Car Sharing Optimization

Regarding the use of approximate methods, both metaheuristics (Glover and Kochenberger, 2006; Duarte et al., 2018) as well as modeling and simulation methods (Law and Kelton, 2000; Macal, 2016) have been employed to deal with ride-sharing and carpooling problems. Among the former, different metaheuristic frameworks have been tested, including: GAs, TS, LS, GRASP, and hybrid methods that aim to combine a few heuristics.

9.5.1 Ride-sharing

Genetic Algorithms: The GAs consist of one of the main approximate methods to solve ride-sharing problems. For example, Herbawi and Weber (2012) studied the dynamic ride-sharing problem. Their objective function includes several dimensions, such as total travel time, distances of the drivers' journeys, total travel time of the passengers, and number of matches. Schreieck et al. (2016) proposed an automated matching algorithm in order to minimize the time to matching rides in a dynamic ride-sharing problem. The proposed methodology is based on matching ride offers and requests. It also uses a smart data structure to increase the calculation speed of matches. The shortest path between request points is created by utilizing the *GraphHopper* open source library (<https://www.graphhopper.com/>).

Local Search Algorithms: Regarding the use of LS, Simonetto et al. (2019) studied the dynamic ride-sharing in order to minimize the duration time of trips. In their work, they recast the ride-sharing problem into a succession of batch processes that combine a linear assignment algorithm, a context-mapping algorithm, and a capacitated vehicle routing problem with pick-up, delivery, and time-windows. The authors used an insertion heuristic to insert new passengers into live rides and developed a LNS to solve this complex variant. Two real-life data-sets are used in order to test their LNS: the New York City taxi data-set and the Melbourne metropolitan area data-set. Apart from proposing an exact approach for solving the ride-sharing problem, Hosni et al. (2014) also introduced an incremental cost heuristic to solve the dynamic version of the problem. In this version, the location of the seekers appears in real-time. For each taxi vehicle, whenever a new request arrives, a minimization problem is solved. This allows to compute the additional cost when including it into the route. The objective is to maximize the total profit –i.e., minimizing the vacant seats, taxi fares to passengers, and the number of vehicles. For solving large instances, Chen et al. (2019) also proposed a savings-based constructive heuristic, which combines the use of ride-sharing with external mobility service providers. Among the positive conclusions

regarding the reduced number of trips and vehicle miles, the authors showed that ride-sharing creates more benefits when the participation is high and when the origins and the destinations of the trips are more spatially concentrated. They achieve up to 31.3% savings in distance-based cost and up to 28.7% reduction in the number of vehicles needed to fulfill the users' travel schedules. Apart from solving a dynamic ride-sharing problem exactly, Lee and Savelsbergh (2015) also proposed a metaheuristic based on neighborhood search and shaking procedures, in order to solve the large-scale instances that the IP model was not able to. Similarly, Naoum-Sawaya et al. (2015) also developed a heuristic to solve real-life instances related to the city of Rome.

Hybrid Methods: Hybridization of metaheuristics has also been employed in the ride-sharing literature. For example, Jung et al. (2016) proposed three different algorithms to solve the dynamic shared-taxi-dispatch problem: a nearest vehicle dispatch (NVD) algorithm, an insertion heuristic (IS), and a hybrid SA. In this problem, passengers on demand are dynamically assigned to empty seats in passenger cars. The NVD simply assigns a passenger to its nearest geographically available vehicle, which is the most commonly used in real-life applications given the need for quick response times. The IS handles real-time passenger requests by considering all feasible vehicles and finds the best available vehicle to assign to a new passenger (which does not have to be the nearest one). Finally, the hybrid SA assigns efficiently and dynamically passengers on-demand to available vehicles. This is done by systematically re-optimizing the assignment of new requests, as well as updating existing schedules in real-time. Two objectives were addressed: (i) minimizing total travel time of passengers, and (ii) maximizing system profit from selectively accepting passengers based on the current schedule. Simulations were conducted to investigate how a shared-taxi system can improve passenger travel –compared to conventional taxi services– by utilizing vehicle resources more efficiently. Apart from proposing an exact approach for solving a ride-sharing problem under travel time uncertainty, Li and Chung (2020) also proposed a hybrid algorithm that combines an extended insertion algorithm with a TS method. The insertion algorithm finds initial feasible routes, which are iteratively improved by the TS. The hybrid TS was able to find near-optimal solutions in shorter computational time, when compared with the exact approach, and to overcome other heuristics' solutions.

Other heuristic methods: Wang et al. (2017a) studied a dynamic ride-sharing problem in which the passenger has the option of accepting or declining the assigned vehicle. They proposed a heuristic algorithm to find stable matches –i.e., those in which no rider and driver, currently matched to others or unmatched, would prefer to be matched together. Similar to Agatz et al. (2011), they used the rolling horizon strategy for dealing with cases in which new trip announcements continuously arrive. Similarly, Najmi et al. (2017) also developed a clustering heuristic to solve a static and dynamic ride-sharing problem to minimize the total traveling distance. They presented a novel clustering heuristic based on both the origin and the destination of users, to solve a large-scale dynamic ride-sharing problem. This algorithm was previously introduced in a static context, being then posteriorly embedded within the rolling horizon strategy, to periodically solve the matching problem as new announcements enter the system. A year later, Li et al. (2018b) have proposed a TS algorithm for solving an

enhanced ride-sharing system with meet points and users' preferable time windows.

9.5.2 Carpooling

Genetic Algorithms: Huang et al. (2015) proposed a genetic-based carpooling matching and routing algorithm to solve a carpooling problem for online systems. In this version of the problem, the driver may pick-up more than one passenger during the trip, respecting capacity constraints, i.e., the number of seats. Then, an efficient matching of drivers and passengers should be provided by the online system. Each passenger is taken by a single driver. The algorithm determines carpooling matches and it is divided into two modules: evolution initialization and genetic evolution. The former transforms the solutions into chromosomes, and the initial population is generated by a distance-based greedy heuristic. The chromosomes are made up of segments, which represent the passengers assigned to each driver. The latter aims to find the optimum carpool route and matching results. In the crossover operator, the segments are combined. The mutation is based on insertions (applied at the segment with the worst sub-fitness) and multiple swaps. A chromosome repair is called when an invalid chromosome is generated. Another use of a GA for solving a taxi carpooling path optimization model was proposed by Ma et al. (2018), where a single objective model was extended to a model with multiple objectives. Apart from minimizing the taxi travel distance, the proposed models aimed to reduce detour distance and cost of passengers, as well to increase the passengers' satisfaction and taxi drivers' income.

GRASP Algorithms: Using a GRASP framework, Santos and Xavier (2015) studied the problem of taxi-sharing combined with carpooling. Carpooling drivers specify their departure point, destination, time departure, and the maximum delay tolerated by the latter. As for taxi drivers, they indicate their current locations as well as the start and end time of their service. The drivers must also fix the price per kilometer, as well as the maximum capacity of their vehicles. Each passenger has a maximum cost that he/she is willing to pay for the trip. The authors' strategy is to solve this dynamic problem by transforming it into a series of static problems.

TS Algorithms: Recently, Cheikh-Graiet et al. (2020) proposed a TS algorithm for solving a dynamic carpooling problem. This dynamic system supports the automatic and optimal ride-matching process between users on very short notice or even en-route, and includes the possibility to drop off passengers at a given walking distance from his destination, in order to increase users' satisfaction. For doing that, the proposed TS employs several original searching strategies developed to make optimal decisions automatically, while allowing transfers and detours. A simulation environment was developed based on actual carpooling demand data from the metropolitan area of Lille, in France. The proposed methodology was able to satisfy a maximum of carpool requests by involving a minimum number of vehicles. This satisfactory performance was achieved by allowing detour and transfer processes.

Other heuristic methods: Yan and Chen (2011) addressed a carpooling problem with pre-matching information, modeled as an integer multiple commodity network flow problem (IMCNFP), and solved by a solution method, based on Lagrangian relaxation and a heuristic for generating the upper bound solution, since the IMCNFP is characterized as *NP-hard*.

The authors were the first to consider the PIM into this problem, which is obtained from previous matching results and can include valuable information, such as carpool partners, the remaining vehicle capacity, and the route/schedule for each previously participating vehicle. The use of PIM aims to reduce inconveniences among commuters since most of the previous users from the same matching would expect similar carpool partners, and drivers would expect any change of their schedule/route to fall within a tolerable range. To test the proposed methodologies, the authors generated a set of 30 instances, based upon data reported from a past study carried out in Taiwan, and they concluded that both the model and solution algorithm were efficient on solving the problem. With the goal of minimizing CO₂ emissions, Bruck et al. (2017) studied the static carpooling and provided two mathematical models and two heuristic-based methods to solve a real application. Su et al. (2019) developed a new hybrid method that combines an artificial bee colony algorithm (Karaboga et al., 2014) with a variable neighborhood search (Hansen and Mladenović, 2014) and a tabu list (Gendreau and Potvin, 2005) to minimize the total distances of all passengers.

9.6 Simulation Methods for Car Sharing Management

The use of simulation for car-sharing management dates back to 70s. For example, motivated by the fuel crisis of 1973 in the U.S., and the scarcity of federal funds for implementing new urban transport facilities, Kornhauser et al. (1977) developed a simulation for assessing the productivity potential of dynamic ride-sharing systems on a hypothetical automated guideway transit network designed for Trenton, New Jersey. Different policies were tested, based on the number of specific origins and destinations that can be served by a vehicle at any one time. For the single-origin to single-destination, the daily average vehicle occupancy improved by 60-120% over the purely non-shared-ride operation. Since then, simulation approaches have been used widely to study car-sharing problems. Among simulation approaches, agent-based and dynamic simulation have been the most frequently used methods to deal with car-sharing issues.

9.6.1 Ridesharing

Agent-based modeling: Regarding agent-based modeling, the taxi ride-sharing problem was addressed in Lokhandwala and Cai (2018) using the New York city fleet as a case study. These authors employed the following implicit objectives: decrease the fleet size, increase the occupancy rate, decrease the total travel distance, and reduce the carbon emissions. The main findings of the paper are that ride-sharing may reduce the service level in suburban areas and that the ride-sharing combining autonomous driving with autonomous vehicles can potentially decrease the fleet size by up to 59%. In their simulations, the total travel distance was decreased by up to 55%. Due to the possibility of full-day operations and the absence of drivers, the use of autonomous vehicles in a ride-sharing system has received increasing attention during the last years. Fagnant and Kockelman (2015) dealt with using shared autonomous vehicles (SAVs) in urban areas. In their work, dynamic ride-sharing

opportunities were included in order to optimize fleet sizing, improve the model's capabilities, and deliver a benefit-cost analysis for fleet operators. These opportunities allow two or more independent travelers to share a single SAV. An agent-based micro-simulation model was proposed to build an SAV fleet to transport those trip-makers from their origins to destinations over a day, which was then modified to allow travelers to access SAVs that are currently occupied or claimed by other trip-makers –i.e., the dynamic ride-sharing system. The proposed model is composed of four modules: (i) the SAV location and trip assignment module; (ii) the SAV fleet generation module; (iii) the SAV movement module; and (iv) the SAV relocation module. The first module assigns waiting travelers to the nearest SAV, prioritizing those who have been waiting longest. In the second module, SAV paths are computed using a Dijkstra-based algorithm to determine the shortest time-dependent route for a SAV to reach each assigned traveler –and his/her final destination. The third module tracks SAV movements of picking-up and dropping-off travelers. Finally, the last module is used to balance the supply-demand over space and time. As expected, the use of ride-sharing mobility is able to improve the model capabilities, hence reducing the average total service time.

The approach by Levin et al. (2017) applied shared autonomous vehicles to ride-sharing and dynamic ride-sharing. There are two main objectives to be minimized: the travel time and the number of SAVs. They also consider a constraint on waiting times. Despite this work does not propose an optimization model itself, a heuristic was created together with an event-based simulator using existing traffic models. The proposed heuristic for dynamic ride-sharing was applied in downtown Austin city, and compared with personal vehicles results from dynamic traffic assignment. A central SAV dispatcher was used to make routes and passenger assignments using centroids as destinations of autonomous vehicles. The paper concludes that some SAV scenarios also increased congestion because there are additional trips made to reach travelers' origins, but the total number of vehicles on the road may be reduced.

Dynamic simulation: Long et al. (2018) were the first authors to propose a stochastic ride-sharing model that addresses stochastic travel times following a time-independent distribution with a positive lower bound. This model was then extended to formulate a stochastic ride-sharing model with time-dependent travel time uncertainty. The model aims to maximize both the total generalized trip cost-saving and the number of matches between drivers and riders. The authors employed MCS in order to estimate the departure time and the minimum trip cost associated with each driving-alone trip and ride-sharing trip. In their work, the time interval is divided into smaller sub-intervals (discretized into many planning horizons), which transforms the dynamic ride-sharing problem into a sequence of static ride-sharing problems. The authors concluded that the travelers' values of time, the unit variable cost of driving, the travel time uncertainty, and the selection of the weights in the objective function have a significant impact on the performance of the ride-sharing systems. Also, a feasible ride-sharing match, based on deterministic travel times, can become infeasible in a stochastic ride-sharing system. Interest readers are referred to the recent survey by Narayanan et al. (2020) for more SAVs applications.

9.6.2 Carpooling

An intelligent route scheme, based on mining GPS trajectories from shared riders to support a carpool service in heavy urban traffic conditions, was proposed by He et al. (2014). In this case, riders with similar preferred routes are grouped by using a GPS-assisted mining approach in order to minimize the driving distance, reduce commute costs, protect the environment, and alleviate urban traffic problems. Drivers' preferences (such as minimizing the total travel costs, the walking distance to make connections, the detour distance to pick-up riders, the social distance, etc.), and the dynamic join-and-leave policy are taken into account. The proposed approach consists of two major subsystems: trajectory mining and carpool routing. The first subsystem processes each user's trajectory log recorded at a rider's GPS device, while the second one runs on the database of extracted (mined) frequent routes. The final route is generated by a pairwise merging process. The authors concluded that increasing walking and detour distance leads to a higher success rate, while excessive detouring will lose carpooling service efficiency. Moreover, the efficiency of the ride-sharing increases with the carpooling size and the response time of finding a candidate driver is unrelated to the total distance of the route, although the decision time of searching qualified passengers is quite related to the route distance.

As we could notice, since the first motivation for adopting a ride-sharing system, addressed by Kornhauser et al. (1977), this problem has become even more complex thanks to new advances in telecommunication and the emergence of mobile technology. Consequently, several solving approaches have been proposed in the literature for solving different variants of these car-sharing problems, which are often enriched by new constraints and objectives. In conjunction with Table 9.1, Figure 9.5 depicts the rate of used solving approaches for each car-sharing activity, according to the previous classification.

When analyzing Figure 9.5, it is noticeable the use of approximated methodologies to solve both the problems related to ride-sharing and carpooling systems. Specifically, for ride-sharing activities, the use of exact approaches is also substantial. However, as mentioned, their use is often limited due to the size of the problem instances, whose particularity transforms its employment unsuitable for solving real-life and large problem instances. On the other hand, the use of heuristics and metaheuristics approaches is the most significant in both cases, being them able to provide high-quality solutions in short computational time as required by such systems.

Apart from the prior classification provided in Table 9.1 and Figure 9.5, we have performed, in Figure 9.6, a new categorization of these studies, including a deeper analysis regarding the addressed optimization objectives for each car-sharing activity.

Among the objectives highlighted in Figure 9.5, minimizing the travel time and/or distance represents the main objective when solving these models. For ride-sharing activities, it is also noticeable the interest in increasing the provider's profit and reducing their respective travel costs. From an operational perspective, several papers aim to reduce the fleet size, which is directly related to other objectives, such as reducing the number of vacant

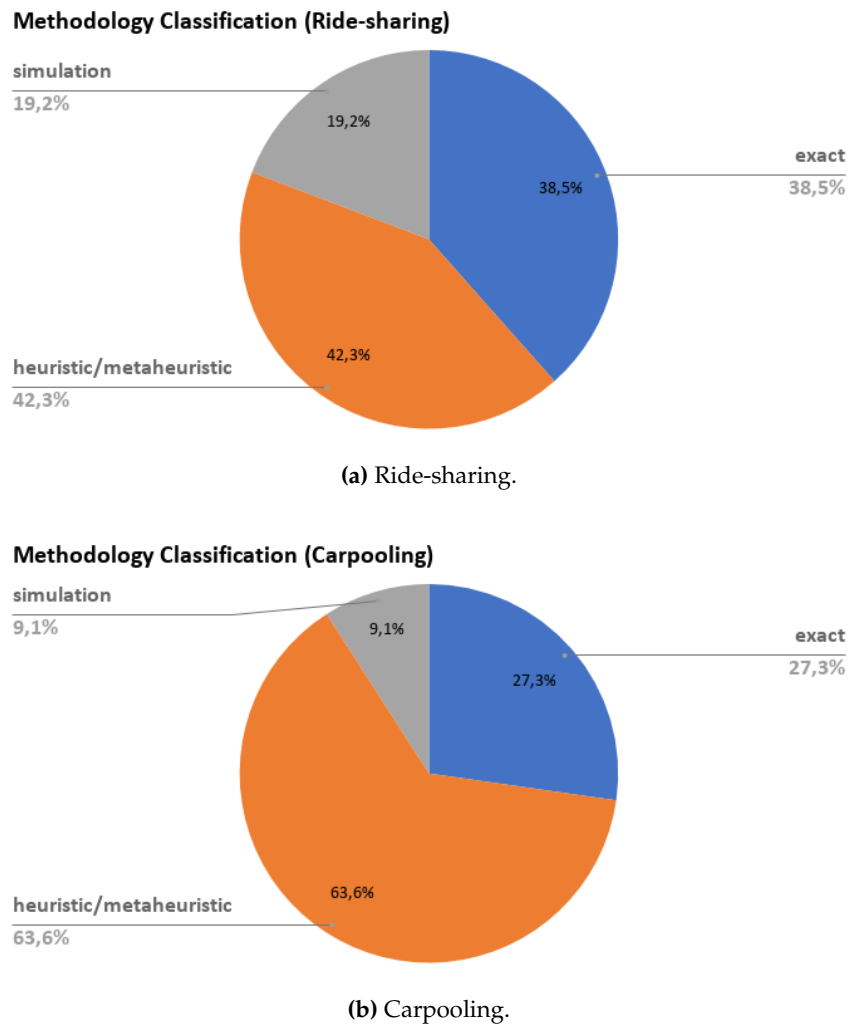
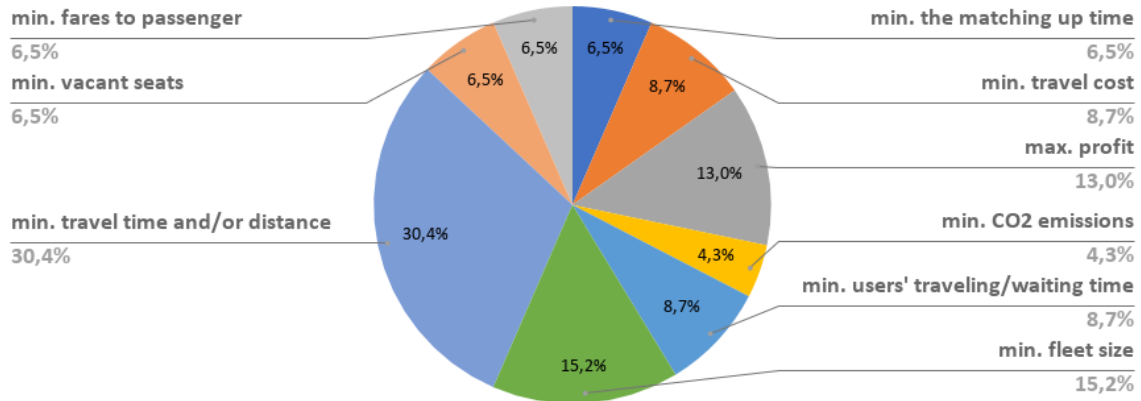


Figure 9.5: The percentage of reviewed papers per solving methodology classification.

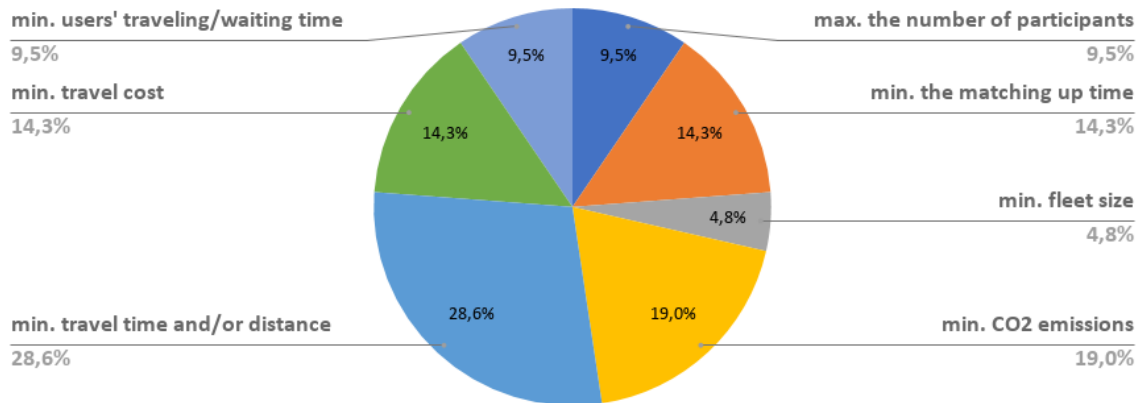
seats and CO_2e . Regarding the latter objective, reducing CO_2e has been considerably established as one of the model objectives in carpooling studies, followed by the minimization of the match-up process between drivers and riders, which is frequently required dynamically and in real-time.

9.7 Performance Analysis of Ride-Sharing Systems

The use of shared transportation systems has led to the improvement of several associated activities in the context of urban transportation. Despite the practical challenges associated with their implementation in real life (such as coordination and synchronization of users, uncertainty, and dynamism of the real-world), ride-sharing and carpooling systems showed to hold the capability of reducing several problems caused by individual transportation. Among them, we can highlight the reduction of congestion on the roads, reduction of vehicle miles traveled, increase of occupancy on vehicles, diminishing both traffic and pollution, reduction of operating costs and fares, and so on.

Optimization Objective (Ride-sharing)

(a) Ride-sharing.

Optimization Objective (Carpooling)

(b) Carpooling.

Figure 9.6: The percentage of reviewed papers regarding the addressed optimization objectives.

In the literature, there exist several studies that address real-cases of ride-sharing and carpooling activities around the world. Some recent studies show how efficient car-sharing systems are able to achieve the goals previously introduced. Most of them depict gains on fleet reduction and its related attainments. For instance, Li et al. (2018a) studied the effects on traffic conditions in the city of Langfang, China, by considering a carpooling system for the existing traffic demand. The proposed system was able to achieve 49% of trip reduction rate and it alleviated the traffic condition in 82.5% of the congested road segments. Moreover, by reducing and alleviating congestion of roads, the carpooling was able to increase the travel speed during peak-hours on most road segments by 5–40%. By analyzing this work, it can be noticed the potential of this system on reducing congestion and, consequently, on improving the locomotion on the roads.

Another example was conducted by Lokhandwala and Cai (2018), for New York City, U.S. This study revealed that autonomous driving in ride-sharing can potentially decrease the fleet size by up to 59%, without a significant increase in waiting time and additional

travel distance. The total travel distance can be decreased by up to 55%, and about 725 metric tonnes of carbon emissions can be reduced per day. Apart from reinforcing previous conclusions about alleviating congestion and reducing vehicles' travel distance, this study further shows how shared-transportation modes can be environmentally beneficial to the population. Similarly, Cai et al. (2019) presented a real case study for quantifying the environmental benefits of ride-sharing taxis in Beijing, China, where the trip information from 12,083 taxis in Beijing was used to identify shareable trips and quantify the potential energy savings and emission reduction. Like previous studies, the use of taxi-sharing throughout the entire day can reduce, annually, fleet vehicle miles-traveled by 33%, save approximately 28.3 million gallons of gasoline and reduce 2,392 tons of CO_2e , among other emissions. However, according to Simonetto et al. (2019), the total number of vehicles employed for ride-sharing services must be limited as a function of the demand, in order to achieve both the traffic and environmental benefits. The latter authors showed how real-time ride-sharing offers clear benefits in terms of the service level, compared to traditional taxi fleets, even considering a partial adoption of the system. In their study, they concluded that approximately only 10% of the current taxi fleet would be needed to meet 96% of the demand in the Melbourne Metropolitan Area, Australia. Accordingly, we can notice how this work supports the efficient use of vacant seats of conventional taxis, being, consequently, able to substantially decrease the number of operating vehicles in metropolitan areas.

In another recent study, Zhang et al. (2020) analyzed the taxi data of Lanzhou City, China. Similar to the previous examples, the use of ride-sharing strategies could reduce the number of taxis by 57% and the travel distance by 44%. Another valuable conclusion is related to the total revenue of each taxi, which is significantly improved when compared to the driving efficiency of the non-sharing mode. Therefore, apart from improving the taxi operation efficiency and save drivers' travel distance, the use of ride-sharing strategies can reduce the passengers' travel expenses and, hence, increase the drivers' travel efficiency. Another example of travel distance reduction is depicted by Wang et al. (2018), which addressed a taxi-sharing case study in Singapore City, Singapore. In their work, the proposed framework was able to achieve not only a reduction in time but also a reduction in travel distance from 20% to 30%.

Based on these studies, we can conclude noticeable environmental benefits, economic impacts, and, especially, transportation issues that can be partially solved when car-sharing activities are employed in big cities. As stated in Simonetto et al. (2019), such shared-transportation modes are also useful in the case of non-monopolistic economies and partial adoption of vehicles, which allows start-ups, small-medium enterprises, and city authorities to embrace their employment for potentially improving transportation and life quality of citizens.

9.8 Challenges Related to Synchronization & Coordination

The following sections will review the main challenges and research opportunities related to the optimization of ride-sharing operations in smart sustainable cities. Figure 9.7 offers

a conceptual map including some of the main keywords that will be further analyzed in Sections 9.8–9.11.

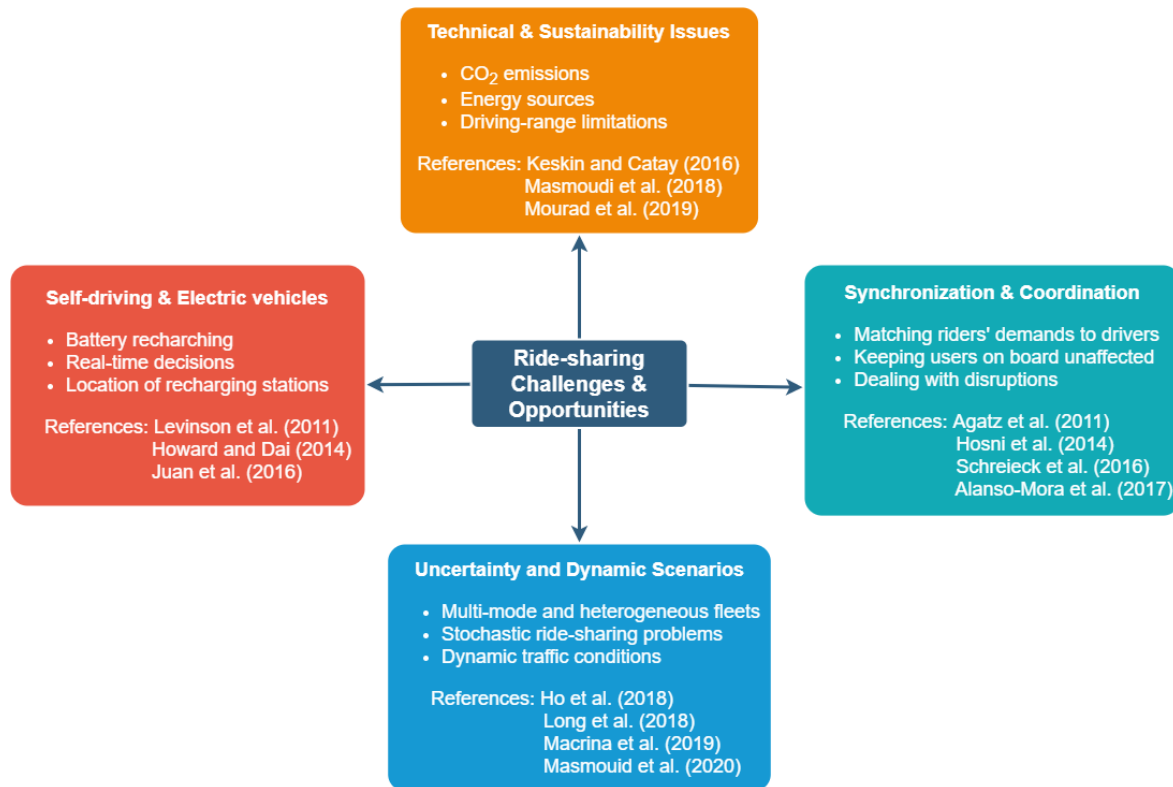


Figure 9.7: Main challenges and research opportunities related to ride-sharing operations.

When dealing with ride-sharing systems, one of the most challenging tasks is how to efficiently match-up a driver offer and a demand from a rider. In a dynamic environment, this matching has to be done in real-time. In general, besides minimizing the driver-passenger matching processing time, the objectives when solving the ride-sharing problem are: (i) to minimize the driving distance, detour distance, commute costs, vacant seats, taxi fares to passengers, and the number of vehicles; and (ii) to maximize the total profit obtained from serving the involved riders –possibly including parcel requests (e.g., Li et al. (2014a))– while indirectly protecting the environment and reducing fuel consumption as well as traffic in urban areas.

As stated by Agatz et al. (2011), this driver-passenger matching process should be largely automated in a dynamic setting. This would allow establishing ride shares in a way that requires minimal effort from the participants. Usually, this process is supported by pre-determined conditions from both the system itself and its users in order to ensure their convenience and satisfaction. It includes, for instance, the consideration of a maximum detour distance, the number of available seats, departure time range, etc. (Schrieck et al., 2016). Accordingly, several papers regarding the use of ride-sharing systems are focused on proposing a solving methodology to match-up drivers and riders on very short notice, or even *en-route*. Most of these methodologies are based on approximate techniques, which are

suitable in the considered application context for being able to provide high-quality solutions in a short computational time. However, some of the studies break the overall problem into sub-problems, which can be exactly solved in a reasonable amount of time. This is the case, for instance, of Hosni et al. (2014). Others make use of simulation approaches for dealing with the dynamism and uncertainty involved in the process (Long et al., 2018). Finally, some works are focused on proposing efficient data structures that allow increasing the matching-up speed (Schreieck et al., 2016).

Once an efficient matching procedure is generated for solving the drivers and riders matching, the resulting routing stage, which considers the newly updated schedules must be resolved. Consequently, when dealing with coordination and synchronization of riders' requests and drivers' offers, the routing stage is transformed into a fully riders-dependent process. Hence, this stage consists of designing optimized routes for attending to different riders' demands, expectations, and objectives without causing any disturbances or interruptions to the drivers who are already on board. Besides the general goal of minimizing the overall transportation costs, this process includes the minimization of waiting times and walking distance for riders, as well as the detour distance for drivers (and, indirectly, for riders who are already on the trip). Since the related routing problem can be seen not only as a vehicle routing problem but also as the dynamic pickup-and-delivery problem (Alonso-Mora et al., 2017), the routing process incorporates different constraints beyond those traditionally considered in the classical vehicle routing problem. Therefore, a significant challenge when addressing the routing stage in ride-sharing systems is the need to generate good solutions fast enough to provide the users with a quality service. In this context, the proposing methodology must be able to incorporate information on trips provided by new users during the planning execution and then maintain the quality service by generating high-quality routes for both the passengers previously assigned in the vehicle and the ones to be incorporated.

From the users' perspective, the challenges vary, for instance, from how to combine ride-sharing with other types of transportation, when ride-sharing is only a part of the users' full trip (Furuhata et al., 2013), to data privacy and trust between drivers and passengers (Svangren et al., 2018). According to the latter study, 80% of the participants had trust issues towards drivers that were materialized as concerns about reliability and privacy. Despite being interested in having some earlier information concerning the other passengers and drivers, most users are still unwilling to give much information about themselves while sharing rides. Therefore, this trade-off between the need for prior information and the reluctant behavior of users to provide them is one additional particularity that makes the use of such systems challenging nowadays. One way to overcome this difficulty is to understand people's attitudes, beliefs, and travel behavior, which can be gathered thanks to the emergence of social media. According to Tang et al. (2019), this valuable information can be used for improving the ride-sharing decisions taken by participants, e.g., by generating dynamic shared-ride plans, improving group queries, optimizing ride matches, and for up to date information notices or other purposes involved in ride-sharing. However, this is one more challenging task due to the need of gathering techniques to extract specific types of

travel-related information.

9.9 Challenges Related to Self-Driving and Electric Vehicles

Figure 9.7 identifies some key challenges in regards to self-driving and electric vehicles. The first challenge is related to the real applicability of autonomous vehicles (AVs). Although the use of AVs represents a breakthrough with the power of changing the modern transportation by transforming it into a more sustainable, safer, and convenient one, self-driving vehicles also bring issues of safety, congestion, fuel, efficiency, and equity (Howard and Dai, 2014). From the human point of view, the use of AVs in public spaces remains an unconvincing way of safe transportation, due to its incapability of dealing and reacting to unexpected or unusual events as a human driver. When considering a real-world application, AVs-related issues are affected by a lot of external factors that change the standard and expected behavior of the involved variables (Levin et al., 2017). For instance, when immersed in a realistic scenario, such as the city centers –in which pedestrians and vehicles share a shared space– decisions must be taken in real-time and dynamically. It might be the case when a pedestrian crosses the road at the wrong time, or when a traffic accident happens. Another example can be described as a road that is blocked off, or even when obstacles are found in the roads. Therefore, in order to solve the resulting problem dynamically and efficiently, the solving methodology must be able to deal with uncertainty, dynamism, and unexpected events during the execution of the planning routes.

When combining the use of autonomous vehicles with an electric-based engine, the ride-sharing problem results in an even more complex scenario to deal with. As pointed out by Juan et al. (2016), the use of electric vehicles (EVs) in smart cities is somehow limited by different strategic and operational challenges. Hence, the second challenge is associated with their ability to cope with the strategic planning point of view. The incorporation and use of EVs in logistics and transportation problems require the consideration of several limitations. For instance, EVs have limited driving-range capabilities, which brings the necessity of installing recharging stations in order to ensure their operation and then to provide an efficient operational plan (Bongiovanni et al., 2019). Consequently, questions such as *how many* recharging stations and *where* they should be installed are raised and must be taken into account.

The last challenge is related to the inclusion of decisions within the operational level. Another implication related to the introduction of EVs in the operational plan is the definition of the best fleet size and their combination (mixed fleet) with conventional vehicles to provide a compelling experience to the market. From the operational planning point of view, we can cite economic (Mourad et al., 2019), charging network (Levin et al., 2017); and (Corlu et al., 2020). Regarding economic aspects, the replacement of conventional vehicles with electric ones is an investment that should be carefully studied by the companies. Subsidies are becoming a usual effort from the government to reduce the acquisition cost of these zero-emission vehicles (Rudolph, 2016; Ma et al., 2017). Moreover, the installation of

recharging stations is another costly investment needed to enable their operation. Regarding charging network issues, they are mainly related to the installation of refueling stations, including how many of them and in which locations. Finally, routing plays a vital role in transportation. In this way, an efficient route-planning, which takes into account the specific mentioned features of EVs, should be provided. This includes the incorporation of recharging stations in the working plan of the routes. Therefore, it is notable that, apart from the advantages of using EVs, their use in smart sustainable cities brings the necessity of redesigning the whole transportation system in order to get its benefits properly.

9.10 Logistics Issues and Uncertainty Scenarios

Figure 9.7 presents some key challenges in relation to logistics and uncertainty scenarios. The majority of ride-sharing studies assume only one mode of transportation, which is based on a homogeneous fleet of vehicles. However, transportation of people or freight is generally carried out by a heterogeneous fleet of vehicles –e.g., vehicles with different capacities, sizes, or energy sources, such as EVs or internal combustion engine vehicles (ICEVs). As indicated by Masmoudi et al. (2020), some alternative fuel vehicles (AFVs), such as flexible fuel vehicles or fuel cell vehicles, use different types of alternative combustibles (e.g., hydrogen propane, ethanol, bio-diesel, liquid natural gas, etc.). Therefore, one crucial feature research in ride-sharing consists of using a heterogeneous fleet of both autonomous and non-autonomous vehicles (EVs, ICEVs, AFVs, etc.), either under static, dynamic, or stochastic scenarios.

It is also possible to use mixed-mode operations, such as the combination of a private transportation fleet with a public one (buses, metro, etc.). In fact, operations that use mixed modes of transportation are quite usual in ride-sharing practices (Macrina et al., 2019). Again, a major challenge of the mixed operations mode is the synchronization of ride-sharing systems with public transit. Cooperation among public and private transportation modes is necessary to complete the requests of users in urban, peri-urban, and metropolitan areas. Thus, for example, when travel times are stochastic a user may be left behind at the transfer point due to a delay in the drop-off time. This will be even more common and critical for an integrated ride-sharing service with the use of public transit that has infrequent service (Ho et al., 2018). Hence, the schedule planner needs to develop robust plans. This can be achieved, for example, by reserving sufficient waiting times at the transfer points. In case of transfers not being realized as planned, it is essential to recover the plan by deploying additional vehicles or making adjustments in the plans of other vehicles. Therefore, several challenges can be found regarding the use of AVs: (i) how to design robust plans involving AVs; (ii) how AVs will interact with the existing modes of transportation; (iii) to what extent will AVs improve transportation efficiency; and (iv) how AVs will benefit from the public/private transportation modes. An interesting research direction is to develop and analyze the impact of using mixed transport modes in a dynamic and stochastic environment. The main challenge is to plan a set of routes by providing the best fleet composed

of two different modes of transport (private and public transit) to satisfy the requests of passengers, where these requests are dynamic and stochastic –including the risk of suffering service disruptions. Some passengers may be transferred from a vehicle to another one on the way to their destination. The main challenge here is that the arrival and departure vehicles should be synchronized. A few papers have considered synchronization aspects (Aissat and Oulamara, 2014; Stiglic et al., 2015). More research could be developed to extend the ride-sharing models by introducing other synchronization aspects, for instance: load synchronization, resource synchronization, and operation synchronization (Drexler, 2012). Another example is to provide flexible driver-to-vehicle systems and multi-depot settings in which the vehicles and the drivers should be synchronized (Ho et al., 2018).

Based on the earlier analysis of the literature, we have observed that only very few studies have been reported for the stochastic ride-sharing problems. More specifically, out of the 29 papers reported in Table 9.1, only a few including Naoum-Sawaya et al. (2015) and Long et al. (2018) consider stochastic elements. The most studied travel times of ride-sharing passengers are assumed to be deterministic (Xu et al., 2015). In real cases, however, there is usually some uncertain information related to travel time. Note that real-world ride-sharing activities are mostly stochastic because the processes are often unpredictable due to changing circumstances, which remain unknown until the process is under execution (Agatz et al., 2012). Similarly, most traditional ride-sharing papers consider only new user requests under a dynamic environment. These studies do not consider other types of events (e.g., accidents, traffic conditions, etc.), which require modifications of existing plans or affect the synchronization of vehicles. As suggested by Ho et al. (2018), disruption management is an important and realistic aspect that should be taken into consideration for any company while planning a set of routes to service its users. In such cases, the already existing routes should be modified to manage the disruption. Hence, the need for developing new models and frameworks to capture these factors of disruption management in the field of ride-sharing. Furthermore, SAVs could set up an interesting mobility option for the passengers (Farhan and Chen, 2018), i.e., SAVs essentially provide a ride-sharing service to travelers. Studying how the SAVs can be managed in such disruption situations can be considered a promising research direction.

9.11 Vehicle Technical Characteristics and Sustainability Issues

Figure 9.7 identifies some key challenges related to vehicle characteristics and sustainability issues. For example, traditional ride-sharing models assume that the service of people is performed by a fleet of ICEVs (Yu et al., 2019) or SAVs/AVs with similar characteristics (Levin et al., 2017): engine speeds, engine displacement, curb weight, frontal surface area, etc. As discussed in Masmoudi et al. (2018), the special characteristics of the vehicle may affect the fuel consumption as well as the CO₂ emissions. Also, the vehicle identification varies according to many physical features, such as curb-weight and vehicle size. Adding to these specifications, we also find variations based on combustion technology. These include engine speed, engine displacement, aerodynamic drag, and engine friction aspects.

If these vehicle aspects are modified or transformed, this may have a remarkable impact on fleet emissions. Additionally, one of the critical aspects that might affect fuel consumption is vehicle aerodynamic durability (Fontaras et al., 2017). Therefore, different vehicle characteristics should be incorporated into the optimization models.

Using a fleet of AVs has received a great deal of attention by researchers until now, due to the importance of this new technology. However, there are some variations in which AV-based systems need to be considered differently. For instance, some privately-owned AVs might be used while their owners do not use them. Therefore, these AVs can be employed on a specific road, which can help to minimize their traffic-related issues compared to ICEVs. In addition, planning recharging stations and maintenance services may need different strategies and techniques, especially as they have multiple charging technologies (Keskin and Çatay, 2016) and the battery may need several hours to be recharged. This can be time-consuming at some re-charging stations (Mourad et al., 2019). Future research considerations in this area include the identification of using a fleet of AVs for people transportation, how AVs respond to passenger mobility needs, and how shared AVs could affect existing routes.

One challenge that arises in most realistic applications of electric AVs in routing problems is that a vehicle of this type may need to frequently recharge its battery to be able to continue the service route, due to their limited battery capacity (Bongiovanni et al., 2019). In addition, the inadequate infrastructure for recharging AVs makes it very difficult to plan the routes of these vehicles. There is a scarcity of recharging stations needed for these vehicles. Also, they are usually not evenly distributed across a certain region, especially when compared to the widely available gas stations on the roads to refuel the ICEVs (Levin et al., 2017). In this regard, effective transportation planning should take into consideration the visits of users, as well as stops in these stations. The need to recharge the battery is frequently encountered during the customary working day. In the context of the ride-sharing problem, not taking the recharging requirements beforehand in planning the service route may cause service disruption due to a shortage in energy, and possible violation of the problem constraints –e.g., the visiting time windows and/or the maximum ride time of users. Such violations can largely lead to the dissatisfaction of customers, which impacts on the overall service quality and breaks one of the main conditions of the ride-sharing. Moreover, to decide when the AVs should be recharged during the planning of routes it is necessary to develop a new realistic energy-consumption function that takes into consideration the characteristics of these vehicles (Corlu et al., 2020). This can be based on the consumption model function developed for the ICEVs or EVs with drivers developed in several works (Masmoudi et al., 2020).

A recent trend in vehicle routing and green logistics is considering environmentally friendly processes in all aspects of the transportation process. Specifically, the reduction of CO_2e is a major concern. In this context, a relevant challenge in the field of green vehicle routing problem (GVRP) is the pollution routing problem (PRP), in which the minimization of energy and CO_2e emissions are widely studied (Demir et al., 2012; Demir et al., 2014). Unlike the traditional objective function for ride-sharing that tries to minimize

the total traveling distance (Wang et al., 2019), operational cost (Alonso-Mora et al., 2017), travel time (Jung et al., 2016), or maximizing the total profit (Yu et al., 2018), future research can consider minimizing the total required energy based on the vehicle characteristics, the environment, the speed of the vehicle, and the traveling distance. In addition, existing ride-sharing models can be extended into multiple-objective ones by introducing dimensions related to profit, operational costs, environmental impacts, etc. So far, the most studied ride-sharing problems consider a single objective function (Wang et al., 2019), while only a few multi-objective ride-sharing studies have been reported (Yu et al., 2019).

9.12 Hybrid x -Heuristics and Agile Algorithms for Ride-Sharing Problems

Regarding the existing approaches for solving ride-sharing problems, we observed that the choice of metaheuristics is becoming increasingly popular. For example, out of the 29 studies presented in Table 9.1, only 5 of them focus on proposing only an exact approach, whereas 9 of them propose a heuristic method in conjunction with an exact method, and 14 of them provide a heuristic/metaheuristic method and/or other solving approaches (e.g., simulation techniques). Despite the importance of using exact solution methods for solving these problems to optimality, the use of such methodologies is often limited by the size of the problem instances or proposed only for validation purposes. This is due to the fact that most ride-sharing problems are *NP-hard*, large-scale, and contain difficult constraints imposed by real-life operations. In our view, future strategies for ride-sharing optimization should consider the following aspects: (i) the development of solving methods that use updated information to cope with stochastic and dynamic ride-sharing variants; (ii) the development of new dynamic and stochastic frameworks and techniques to capture different events (e.g., accidents, failures, etc.) that can happen in the existing route planning; (iii) the development of *agile optimization* (AO) algorithms able to provide real-time solutions.

Regarding stochastic variants of the ride-sharing problem, the combination of metaheuristics with simulation, also known as *simheuristics* (Juan et al., 2018), can be an effective methodology. Some recent applications of simheuristics can be found in areas as diverse as waste collection management under uncertainty (Gruler et al., 2017), arc routing problems with random demands (Gonzalez-Martin et al., 2018), flow-shop scheduling problems with stochastic processing times (Gonzalez-Neira et al., 2017), project portfolio management under uncertainty (Panadero et al., 2018), or inventory routing problems with stochastic demands (Gruler et al., 2020). Similarly, when dealing with ride-sharing variants under dynamic conditions (e.g., traffic conditions that evolve over time), one promising approach is the hybridization of metaheuristics with machine learning methods, also called *learnheuristics* (Calvet et al., 2017). Recent applications of learnheuristics to different vehicle routing problems under dynamic conditions can be found in Calvet et al. (2016a) and Arnau et al. (2018).

Despite being able to generate high-quality solutions for a range of optimization problems, traditional solving methodologies, such as exact methods, metaheuristics, and simulation techniques, might not represent the most suitable approach when a real-time solving limit is imposed as a hard operational constraint of the associated *NP-hard* problem. In order to deal with this limitation, the concept of agile optimization has arisen as a new optimization and decision-making tool for solving optimization problems in real-time. As mentioned, dynamic ride-sharing requires the dealing of new information dynamically in real-time, often during the plan in execution, i.e., *en-route*. This information includes the trip information, which leads to the necessity for several taking real-time decision making. Hence, due to these likely continuous changes, a re-optimization of the system is required each time new data should be incorporated into the model.

AO refers to the massive parallelization of biased-randomized (BR) algorithms, which are extremely fast in execution, easily parallelizable, flexible, and require the fine-tuning of a few, or even just a single parameter. In the BR techniques, a biased (non-symmetric) randomization effect is introduced into a heuristic procedure by using a skewed probability distribution. This simple mechanism extends a deterministic heuristic –which is extremely fast in execution, even for large-scale optimization problems– into a probabilistic algorithm without losing the logic behind the original heuristic (Ferone et al., 2019). The core idea of AO is to run several hundred or even thousands of threads in a concurrent way, being each one an execution of a BR heuristic. As a result, many alternative solutions are generated in the same wall-clock time as the one employed by the original heuristic –some of them outperforming the one generated by the original heuristic– and the best solution is chosen. Therefore, in addition to the advantage of finding reasonably good solutions in real-time, the use of AO algorithms for solving (dynamic) ride-sharing problems can be seen as a useful approach for solving this type of problems in which new information arrives all the time. In summary, AO algorithms represent a new paradigm in the design of optimization algorithms, which follows the following principles: (i) extremely fast execution, thus providing real-time decision support; (ii) easy to implement and run using parallelization techniques; (iii) flexibility to deal with different T&L problems and variants; (iv) parameterless, hence avoiding complex and time-costly fine-tuning processes; and (v) specifically designed to run iteratively every few seconds or minutes –hence allowing for high-frequency re-optimization– as new streams of data arrive in a dynamic and connected environment. This novel AO approach represents a breakthrough with respect to traditional optimization, simulation, and machine learning methods, which typically require long computation times –and, therefore, cannot deal with present and future T&L scenarios using unmanned and self-driving vehicles, which are characterized by their dynamism and uncertainty. Notice that AO works in an environment of dynamic (constantly changing) conditions, whereas traditional optimization tends to oversimplify these important aspects of the real world. Traditional optimization frameworks are limited when dealing with real-time coordination and optimization needs in current and future T&L applications in smart sustainable cities. This is especially the case when electric, unmanned, and connected/self-driving vehicles are considered in ride-sharing and carpooling activities. Using a scale from 1 (low performance)

to 5 (high performance), Figure 9.8 shows a comparison of multiple analytical methodologies in terms of dimensions such as: (i) capacity to provide optimal values (exact methods excel here); (ii) computational time required (both heuristics and agile algorithms show the highest speed levels, offering real-time solutions); (iii) flexibility to model real-life situations (simulation excels here); (iv) capacity to deal with uncertainty scenarios (simulation and simheuristics show a superior performance here); (v) capacity to deal with large-scale problems (heuristics, metaheuristics, and agile algorithms surpass the others); and (vi) capacity to deal with dynamic environments (learnheuristics, heuristics, and agile algorithms excel in this one).

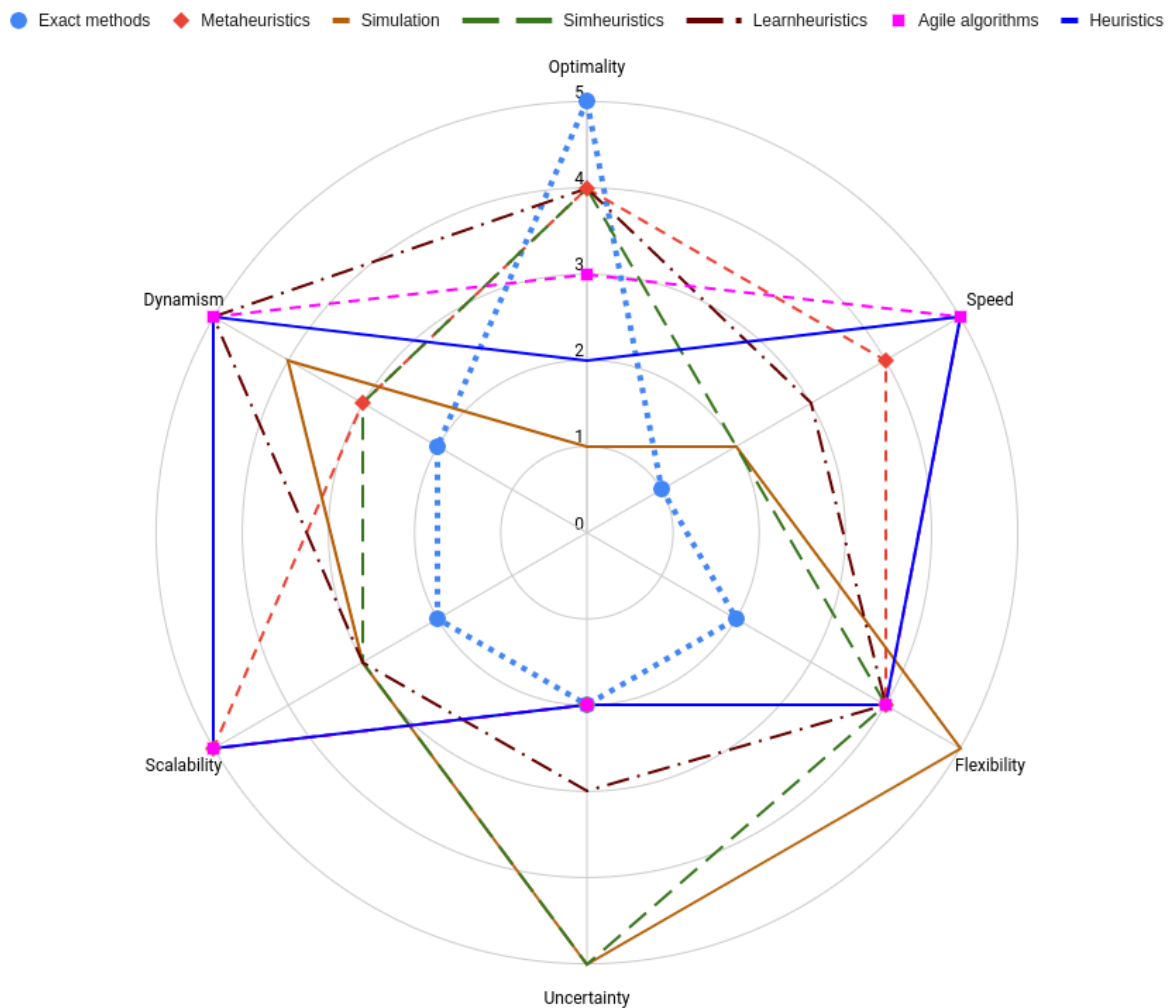


Figure 9.8: Multi-dimensional comparison of different analytical approaches.

9.13 Conclusions

From the trends analyzed in this work, ride-sharing operations in smart and sustainable cities are expected to continue growing over the next few years. Therefore, policy-makers should consider how to optimize these operations to provide timely and efficient service to citizens and, at the same time, minimize important aspects such as the impact of the

mobility of people in the environment and social activity inside the city. In this regard, our review confirms the existence of studies that show clear benefits of ride-sharing and carpooling practices in urban areas, such as: (i) a reduction in the overall cost of mobility systems, measured in travel time and in energy consumption; and (ii) a notable reduction in the volume of vehicles circulating in the city, which could also lead to lower levels of CO_2e .

However, there are still several aspects that must be taken into account by authorities when designing, developing, and, especially, implementing these systems for real-life applications. In this regard, we hope that this article sheds light on these issues. For this purpose, a review of the existing literature on the ride-sharing and carpooling optimization problems has been presented. We expect to facilitate the identification of problems and the analysis of alternatives based on experiences in other urban areas. Likewise, the most relevant studies in the field have been classified according to the analytical methodology used, that is, exact methods, metaheuristics, or simulation, which can help with the decision-making process considering different environments (including uncertainty). The particularity and dynamism in real-time of these problems make them especially difficult to adapt to real cases. In this case, some authors, for example, Borcuch (2016), point out that, from a government point of view, the biggest challenge for any city in adopting these shared modes of transport is how to find a balance between adopting these platforms and regulating them in the name of safety and responsibility.

In order to reinforce the analysis of alternatives and the decision-making process, our study has also identified the main challenges and research opportunities related to the optimization of shared trips. In this way, it is also expected to serve as a handbook for policy-makers that helps navigate towards a more sustainable (environmentally, socially, and economically) city paradigm. This may include the analysis of the vehicles' capacity, the boosting of multiple charging technologies, the creation of charging stations, or the design and planning of more sustainable routes, among others. In terms of main challenges, this chapter illustrates those challenges related to synchronization and coordination issues, as well as the increasing inclusion of electric and autonomous vehicles in our modern urban, peri-urban, and metropolitan areas. In terms of research opportunities, the chapter analyzes the research opportunities associated with the inclusion of heterogeneous vehicle fleets, dynamic scenarios, conditions of uncertainty, technical characteristics of the vehicle, and energy and sustainability issues (for example, type of fuel required and level of carbon emissions), etc. With all this, it is intended that the managers of these areas are aware of the changes that the incorporation of these practices in their cities implies, the improvements it will bring, as well as the resources that will be necessary for their implementation.

Finally, the document goes a step further and presents new approaches to deal with resource optimization problems in carpooling in real life, which must take into account random events and dynamic traffic conditions. To address these issues, the need to develop new hybrid approaches that combine metaheuristics with simulation and/or machine learning methods is analyzed. Also, the article highlights the concept of agile optimization algorithms, which allow generating good-quality solutions in real-time (even less than a second) and recalculating them every few minutes as new data becomes available.

Chapter 10

Conclusions, Future Research and Outcomes

10.1 Final Conclusions

In this thesis, several T&L COPs which arise in the context of smart and sustainable cities were studied. These problems are inherent from diverse application contexts, such as transportation, humanitarian logistics, the retailing industry, production systems, telecommunication, and health care, which implies each of them requiring distinctive solution techniques capable to cope with their particularities. In this way, based on the defined research objectives in Section 1, the following conclusions are written down:

- *Realistic business and industrial optimization problems from the integrated logistics, especially those belonging to the transportation of goods, are formally addressed and defined:*

From transportation to telecommunication, this thesis has addressed different COPs. The majority of related research outcomes address the omnichannel vehicle routing problem, which consists of a variant of the classical vehicle routing problem. The OCVRP is a two-echelon VRP with simultaneous pick-up and delivery, derived from many retailing and transportation systems. From this problem, its stochastic variant has been also studied for the first time in literature, in which travel times between visiting points are modeled as random variables following a specific probability distribution. Still in transportation, the vehicle routing problem with optional backhauls was discussed. This another variant of the classical VRP is characterized by two types of decisions: first, a set of customers must be visited for delivering goods, and later, a set of customers must be visited for picking up returnable transport items. The RTIs are optionally collected by the vehicles, incurring penalization costs. The visiting of picking-up nodes is assumed to be performed one day after the delivery is done for collecting the RTIs employed the day before.

The OCVRP has been also addressed in the context of humanitarian logistics, where relief operations commonly require immediate decision-making. In this regard, the OCVRP has been re-formulated by considering a set of pharmaceutical laboratories and first-aid locations as visiting points (former retail centers and customers, respectively), in which a fleet of unmanned aerial vehicles is employed for delivering and

picking up medical supplies due to the resulting poor transportation infrastructure after a disaster event. Moreover, due to the application context, an additional constraint that imposes the generation of feasible solutions in real-time has been incorporated into the new model.

Another COP that has been identified is the multi-period product display problem with dynamic attractiveness, whose scope belongs to the retailing industry. Although this problem is aimed at the fashion industry, the problem of selecting, over a multi-period horizon, the most attractive configuration of products to be displayed in a limited space can be found in many other sectors, including online stores which dispose of limited digital space to expose the offered products.

From production systems, a hybrid flow-shop problem has been considered conjointly with routing decisions. In other words, this system aims at producing different types of products at a central facility (depot) through an HFS environment, while finished products are grouped into batches and delivered to their destinations through a VRP environment. Since a single capacitated vehicle is available at the depot at the beginning of the production, several trips need to be possibly performed in order to deliver all the requested goods. This combined problem, HFS-VRP, is found in several realist cases in which the company is responsible for both the production and delivery activities, such as optical and pharmaceutical laboratories.

In telecommunications, two related problems have been addressed in this thesis. Firstly, the facility location problem for the internet of vehicle scenarios was presented. In this problem, moving cars need to exchange data with a different range of devices through roadside units that are placed at fixed locations on the street. The connection of cars to deployed (in-service) RSUs must be done quickly and efficiently to ensure a target quality of service of the system. Posteriorly, this problem has been extended into a multi-period variant in which demands vary over time, thus, requiring deactivating and activating RSUs to meet the dynamic needs of the system. Consequently, the dynamic system requires the re-optimization of the system as new data is gathered.

The COVID-19 crisis in 2020 has changed the world. With that, an important challenge in the metropolitan area of Barcelona, Spain, has emerged: how to organize the daily collection of multiple sanitary components from individual homes so they could be transported to the assembling centers and, later, distributed to the different hospitals in the area. This problem has been modeled as different rich VRP variants, including the team orienteering problem, open vehicle routing problem, and the rich pick-up and delivery problems. Since the problem characteristics were different each day, the most suitable model was used to solve the corresponding problem. The goal was to design daily routing plans to pick up the maximum number of items in a limited time, hence reducing the drivers' exposure to the virus.

- *Efficient solution algorithms, based on biased-randomization of heuristics, metaheuristics, and simheuristics, are proposed and developed for solving the identified optimization problems:*

For solving the problems described in previous sentences, different solution methodologies have been proposed and developed. The core of all these methodologies regards the use of biased-randomized heuristics. As explained, BR algorithms make use of skewed probability distributions to bias the behavior of the original deterministic heuristics, being them able to provide solutions with an interesting balance in respect to the solution quality and computational time.

Different metaheuristic approaches have been developed by ‘wrapping’ these BR heuristics, such as the multi-start, greedy randomized adaptive search procedure, iterated local search, and variable neighborhood descent. The MS strategies are simple metaheuristic frameworks that are based on the generation of multiple solutions and the selection of the one with the better objective function value. The GRASP, on the other hand, relies on the use of a restrict candidates’ list that restricts the possible candidates to be selected by a constructive heuristic, besides applying a local search mechanism into the generated solution in order to improve its quality. The ILS goes further by employing, apart from a local search mechanism, a shaking/perturbation procedure to perform reasonably large movements in the solutions, thus allowing the escape from local optima, and also, an acceptance criterion. Finally, the VND operates by performing systematic deterministic changes of neighborhood structures within a local search mechanism, contrasting with most local search heuristics that use only one neighborhood structure to explore the space of possible solutions.

Another class of solution methods that have been developed to solve some of the addressed problems are the simheuristics. As mentioned, despite being able to generate high-quality solutions in a reasonable amount of time for a vast of COPs, the metaheuristics approaches are not the most suitable solution method alternative to cope with stochasticity. In this way, the simheuristics combine the use of heuristics/metaheuristics with simulation in order to deal with stochastic COPs. These methodologies are relatively easy to implement, scalable, and capable to generate solutions with different trade-offs between solutions’ cost and reliability under stochasticity, thus providing decision-makers with risk or reliability analysis criteria regarding the stochastic solutions.

- *An original approach –agile optimization– is proposed to cope with real-time decision-making, such as those required in relief operations or telecommunication systems:*

All the aforementioned solution methods are efficient approaches that relies on sequential and iterative execution. Consequently, the more iterations are necessary to find an effective solution for a COP, the longer this process might take to process it. In this way, the novel concept of agile optimization has been introduced. AO combines the use of biased-randomized heuristics, which are extremely fast in execution, with parallel computing. As a result, several hundred or thousands of threads are executed in parallel, being each a BR heuristic responsible for generating a solution, and the best-found one is returned at the end of the process. By generating all these solutions

in the same wall-clock time as the one employed by the original heuristic, this solution methodology results in an efficient approach to deal with dynamic and large scale *NP-hard* COPs in real-time. However, it is worth to be noticed that the lack of resources can destroy the idea behind agile optimization, in case the available resources are not enough for running all these threads in parallel without getting possible overheads.

- *A series of computational and numerical experiments are conducted to evaluate the performance of the developed algorithms when applied to realistic and large-sized problem instances:*

Several COPs and solving approaches have been addressed in this thesis. Accordingly, different existing benchmark instances have been employed to investigate the efficiency of developed solution methods. For the cases in which no benchmark instances exist to represent the problem model under study, e.g., some of the problems addressed in Chapters 5, 6 and 7, corresponding sets of instances have been generated. Some of these new instances are based on existing benchmarks, while the extra information inherent of the new problem variant has been incorporated to represent the current problem formulation.

The majority of the developed solving methodologies are fed up with at least one parameter. It requires proper fine-tuning of them for guaranteeing the effective performance of the solution methodologies. Therefore, an earlier stage when applying these approaches to solve the addressed COPs regards this experimentation stage. Accordingly, we have employed different strategies, including the methodology proposed in Calvet et al. (2016b), which is based on a general and automated statistical learning procedure.

With the parameters setting defined, extensive computational experiments have been conducted to evaluate the performance of the developed and proposed solving approaches. Different statistical analyses have been performed for validating and supporting the taking of decisions regarding the most efficient solution approaches to tackle the different addressed COPs. Among them, it is highlighted the use of the ANOVA (Analysis of Variance) test for recognizing significant differences in the experiment results when comparing with different approach alternatives, followed by the Fisher or Tukey tests to identify the most efficient approaches.

- *Managerial insights and conclusions about the potential advantages of using the developed algorithms in complex and real-life decision-making processes are drawn:*

The deterministic and stochastic COPs that have been addressed so far are of high complexity and large-scaled. Therefore, their corresponding solution approaches are desired to go beyond the ability to generate high-quality solutions regarding only the optimization cost, in order to foment the development of sustainable smart city logistics. As presented, the use of biased-randomized metaheuristics approaches to solve these problems from smart city logistics is the core of this thesis, being them lately enriched to cope with uncertainty and real-time decision-making. The use of these

approaches has proved their capability of generating near-optimal solutions –in order of seconds or minutes– for the majority of the addressed problems. For the VRPOB (Section 3.2), for instance, an average gap of 0% was achieved by a BR ILS approach when compared with existing optimal solutions. In the case of OCVRP (Section 4.1), an average gap of 13% has been achieved by a biased-randomized multi-start approach, within a maximum running time of 5 seconds. For one of the testing scenarios, this solution method was able to obtain a gap of only 6% against the best-known solutions, contrasting with the alternative solution approach that requires 1,197 seconds, on average, to generate them. Regarding the MPPDPDA (Section 5.1), it has been noticed that both the BR versions of the proposed ILS and GRASP approaches have been responsible for achieving solutions up to 3% more attractive, on average, than the greedy methodology –which can be assimilated to the human behavior. For some instances, the gap achieves 5%, while an average execution time of up to 40 seconds is required for generating feasible and enhanced solutions for the problem. In the case of HFSVRP (Section 6.1), the proposed BR-VND metaheuristic was tested against a set of solutions generated by a MILP model and also against a set of computed LB values. As discussed, for the set of small-sized instances, the solutions found by the MILP model, within one hour of execution, are only 8.22% better, on average, than the BR-VND results. It is worth saying that the BR-VND solutions, on the other hand, have been obtained within few seconds, being the largest instance (from the set of small-sized instances) executed for 9 seconds, which represents the maximum execution time allowed to an instance at this experimentation stage. Besides, one optimal and one better solution were found BR-VND. As a general conclusion regarding the MILP model, their use is not suitable in practical environments since it was able to find solutions for only 13 out of the 54 instances, considering one hour of maximum running time. On the other hand, the BR-VND was able to find feasible and high-quality solutions in order of seconds for the complete set of small-sized instances. Regarding the large-sized instances, the BR-VND found solutions only 18% more costly from the calculated LB when considering a specific combination of initial solution procedure, loading strategy, and solution representation. Based on these analyses, it has been noticed the capability of these algorithms to generate high-quality and competitive solutions for complex and large-sized problems without requiring extensive computational time, contrasting with exact approaches.

Regarding the use of agile optimization strategies, it has been certified their capability of generating high-quality solutions for large-scale COPs in real-time. These needs arise in environments with high dynamism or relief operations, for instance, in which good decisions must be taken quickly. Specifically, for the OCVRP in the context of humanitarian logistics (Section 4.1), where an extra hard constraint of generating solutions in real-time is added into the problem formulation, the solutions for small-sized instances are only 7% more costly, on average, when compared with the optimal solutions. When comparing with large-sized instances, the solutions obtained by the

AO approach are only 17% more expensive, on average. Still, for one of the problem scenarios, a gap of only 9% has been achieved. By noticing that this approach took an average CPU time of 0 seconds to generate these solutions with a reasonably small gap, compared with alternative approaches from the literature that require up to 5,321 seconds to generate their corresponding solutions, it is noticeable the capability of such strategies to deal with real-time decision-making. In the case of the single-period IoVP (Section 7.1.2.1), where moving cars are connected with RSUs, and this connection/assignment must be updated as the cars move to provide an efficient connection between them, the AO was similarly able to find the solutions only 2.6% more expensive than the optimal ones in 0 seconds, on average, against the 15,171 seconds required by an exact solver. For the enriched variant, the MPIoVP (Section 7.1.2.2), the proposed dynamic approach, which is embedded into an AO framework, is able to generate solutions 34% cheaper, on average, when compared with a 'static' approach. Moreover, the AO strategy requires only 0.8 more seconds than the alternative solution method, on average, but can generate significantly better results. Therefore, it is clear the efficiency of this strategy when real-time decision-making is applicable and necessary. Moreover, AO strategies, by being fast and efficient solving approaches, have proved to be satisfactory methodologies to tackle dynamic problems where new information is often gathered by the system, hence, requiring its re-optimization frequently over time.

Simheuristics consist of another decision-support tool that has been proved to be satisfactory to solve stochastic COPs. These methodologies are able to provide results within a few seconds or minutes, even for large-scale problem instances, like those addressed in real-life systems. Certainly, their main particularity regards their ability to account for uncertainty smartly. For the stochastic OCVRP (Section 3.1.2.2), for instance, it has been shown that the simheuristic approach can generate better solutions for the stochastic scenario when compared with the solutions returned under the deterministic assumptions when placed in the stochastic environment. Moreover, it has been depicted that the best solutions for the deterministic case, in terms of cost, are often not reliable under stochasticity, which means that these two measures are highly conflicting. In other words, these solutions present a high rate of failing –being not feasible– when considering uncertain inputs. The latter refers to the ability of simheuristics to generate solutions with different trade-offs between solutions' cost and reliability under stochasticity, therefore, providing decision-makers with risk or reliability analysis criteria regarding the stochastic solutions.

- *Finally, an in-depth study in the context of new modes of passenger transport, such as carpooling and ride-sharing systems, is conducted. As a result, new lines of research are presented:*

With a large number of people living in urban and metropolitan areas, and the increasing number of activities derived from e-commerce and on-demand economy, urban centers are suffering from many problems, such as traffic congestion and pollution. One way to overcome these problems, which are considerably caused by the use of

light-duty vehicles, is through increasing the efficiency of transport systems. Monetary, environmental, and social costs are associated with the low occupancy of personal vehicles, whose effective use holds the capability of mitigating related problems, such as greenhouse effects and global warming. In this way, a survey was conducted in the context of shared mobility systems, such as carpooling and ride-sharing services. These car-sharing systems involve different optimization problems with diverse objectives, such as minimizing travel time/distance, fares for passengers, the number of vacant seats, CO_2e emissions, users' waiting/walking time, or maximizing profit for drivers, for example. These problems are naturally dynamic since service providers match travelers with similar itineraries and time schedules to share a ride on short notice in a personal vehicle.

10.2 Future Lines of Research

The development of this thesis has addressed many challenges from the operational planning level of supply chains, telecommunication, and automobile-based shared transportation systems. In the following, some of them are summarized:

- One of the main problems on which this thesis is based refers to rich VRPs, such as the OCVRP and VRPOB. However, despite these two VRP variants taking into account the sole economic aspects regarding the operation costs in the optimization process, the environmental aspect can be also introduced. Currently, there is a great scientific and social appeal for sustainable alternatives in T&L activities. To this aim, these problems can be further extended in order to ponder environmental aspects, resulting in not only smart but sustainable solutions for smart cities. In the same line, the use of electric vehicles in these systems holds the capability of reducing monetary costs and environmental/social impacts in urban centers. However, relevant challenges emerge, such as their limited batteries' driving range, high recharging times, and the scarcity of recharging stations.
- Another different application context approached in this thesis refers to telecommunication systems. In this case, virtual resource allocations, as those coped in Chapter 7, consist in dynamic environments where a large amount of data is frequently gathered and processed. Consequently, their related COPs must be re-optimized as new information is incorporated into the system, and the solution methodology quick and efficient enough to properly deal with these particularities. To this aim, the proposed AO strategies were tested in these systems, which are modeled as well-known FLPs. Therefore, new lines of research in this field regard the proper modeling of these systems in order to consider telecommunication parameters such as latency, quality of service, energy consumption, coverage, downlink and uplink rates, among others.
- Regarding the use of shared transportation systems, such as ride-sharing and carpooling, many practical challenges associated with their implementation in real life have

been found. Coordination and synchronization of users, uncertainty, and dynamism of the real world represent the most relevant challenges when dealing with these activities. In this way, different research opportunities can be listed. From the methodology point of view, these systems require fast solving approaches capable enough to deal with their dynamism while optimizing a series of different objectives, such as minimizing the number of vacant seats, taxi fares to passengers, and operational costs for drivers, for instance. For this purpose, the use of agile optimization techniques is highlighted, which allows generating good-quality solutions in real-time, and recalculating them every few minutes as new data becomes available. Another related research opportunity refers to the use of hybrid approaches, such as simheuristics or learnheuristics, that combine metaheuristics with simulation and machine learning, respectively, to account for random events and dynamic traffic conditions. However, these intelligent algorithms need to provide a good balance in terms of efficiency and execution time. More future lines of research cover the inclusion of energy and sustainability issues (for example, type of fuel required and level of carbon emissions), which is related to the challenges first mentioned.

10.3 Research Outcomes

In the following, the complete scientific production that has been developed, including articles published in different ISI-JCR/Scopus indexed journals and proceedings of international peer-reviewed conferences, and those under peer-reviewing process, are enumerated.

10.3.1 ISI-JCR Indexed Papers

10.3.1.1 Published

1. **Martins, L. C.**; Hirsch, P.; Juan, A. A. (2020): [Agile optimization of a two-echelon vehicle routing problem with pickup and delivery](#). *International Transactions in Operational Research* (indexed in ISI SCI, 2019 IF = 2.987, Q2; 2019 SJR = 1.018, Q1). ISSN: 0969-6016.
2. Londoño, J. C.; Tordecilla, R.; **Martins, L. C.**; Juan, A. A. (2020): [A biased-randomized iterated local search for the vehicle routing problem with optional backhauls](#). *TOP* (indexed in ISI SCI, 2018 IF = 0.892, Q3; 2013 SJR = 0.600, Q2). ISSN: 1134-5764.
3. Bayliss, C.; **Martins, L. C.**; Juan, A. A. (2020): [A two-phase local search with a discrete-event heuristic for the omnichannel vehicle routing problem](#). *Computers & Industrial Engineering* (indexed in ISI SCI, 2019 IF = 4.135, Q1; 2018 SJR = 1.469, Q1). ISSN: 0360-8352.
4. **Martins, L. C.**; de la Torre, R.; Corlu, C. G.; Juan, A. A.; Masmoudi, M. A. (2021): [Optimizing ride-sharing operations in smart sustainable cities: Challenges and the need for agile algorithms](#). *Computers & Industrial Engineering* (indexed in ISI SCI, 2020 IF = 5.431, Q1; 2020 SJR = 1.315, Q1). ISSN: 0360-8352.

5. **Martins, L. C.**; Gonzalez, E.; Hatami, S.; Juan, A. A.; Montoya, J. (2021): [Combining Production and Distribution in Supply Chains: the Hybrid Flow-Shop Vehicle Routing Problem](#). *Computers & Industrial Engineering* (indexed in ISI SCI, 2020 IF = 5.431, Q1; 2020 SJR = 1.315, Q1). ISSN: 0360-8352.
6. **Martins, L. C.**; Tarchi, D.; Juan, A.; Fusco, A. (2021): [Agile Optimization for Real-Time Facility Location Problem in Internet of Vehicle Scenarios](#). *Networks* (indexed in ISI SCI, 2020 IF = 5.059, Q1; 2020 SJR = 0.977, Q1). ISSN: 1097-0037.
7. **Martins, L. C.**; Tordecilla, R.; Castaneda, J.; Juan, A.; Faulin, J. (2021): [Electric Vehicle Routing, Arc Routing, and Team Orienteering Problems in Sustainable Transportation](#). *Energies* (indexed in ISI SCI, 2020 IF = 3.004, Q3; 2020 SJR = 0.598, Q2). ISSN: 1996-1073.
8. Tordecilla, R.; **Martins, L. C.**; Panadero, J.; Copado-Méndez, P. J.; Perez, E.; Juan, A. A. (2021). [Fuzzy Simheuristics for Optimization Problems in Transportation: dealing with stochastic and fuzzy uncertainty](#). *Applied Sciences* (indexed in ISI SCI, 2020 IF = 2.679, Q2; 2020 SJR = 0.435, Q2). ISSN: 2076-3417.

10.3.1.2 Under Peer-Review

1. Cesarano, L.; Croce, A.; **Martins, L. C.**; Tarchi, D.; Juan, A. A. ([u.r.](#)): An Agile Optimization Approach for the Multi-Period Internet of Vehicles Problem. *IEEE Access* (indexed in ISI SCI, 2020 IF = 3.367, Q2; 2020 SJR = 0.587, Q1). ISSN: 2169-3536.
2. Bayliss, C.; Juan, A. A.; Panadero, J.; Copado-Méndez, P. J.; **Martins, L. C.** ([u.r.](#)): A Learnheuristic Algorithm for the Temporary Facility Location and Queuing Problem during Epidemic Events. *International Transactions in Operational Research* (indexed in ISI SCI, 2020 IF = 4.193, Q2; 2020 SJR = 1.032, Q1). ISSN: 0969-6016.
3. Peyman, M.; Copado-Méndez, P. J.; **Martins, L. C.**; Tordecilla, R.; Xhafa, F.; Juan, A. A. ([u.r.](#)): Edge Computing and IoT Analytics for Agile Optimization in Intelligent Transportation Systems. *Energies* (indexed in ISI SCI, 2020 IF = 3.004, Q3; 2020 SJR = 0.598, Q2). ISSN: 1996-1073.

10.3.2 Scopus Indexed Papers

10.3.2.1 Published

1. Marmol, M.; **Martins, L. C.**; Hatami, S.; Juan, A. A.; Fernandez, V. (2020): [Using biased-randomized algorithms for the multi-period product display problem with dynamic attractiveness](#). *Algorithms* (indexed in ESCI; 2018 SJR = 0.276, Q3). ISSN: 1999-4893.
2. **Martins, L. C.**; Bayliss, C.; Juan, A. A.; Panadero, J.; Marmol, M. (2020): [A savings-based heuristic for solving the omnichannel vehicle routing problem with pick-up and delivery](#). *Transportation Research Procedia* (indexed in Scopus, 2018 SJR = 0.601). ISSN: 2352-1465.

3. **Martins, L. C.**; Bayliss, C.; Copado-Méndez, P. J.; Panadero, J.; Juan, A. A. (2020): [A Simheuristic Algorithm for Solving the Stochastic Omnichannel Vehicle Routing Problem with Pick-up and Delivery](#). *Algorithms* (indexed in ESCI; 2019 SJR = 0.358, Q3). ISSN: 1999-4893.
4. Bayliss, C.; Copado-Méndez, P. J.; Panadero, J.; Juan, A. A.; **Martins, L. C.** (2020): [A Simheuristic-Learnheuristic Algorithm for the Stochastic Team Orienteering Problem with Dynamic Rewards](#). *Proceedings – 2020 Winter Simulation Conference (WSC)* (indexed in ISI WOS and Scopus, 2019 SJR = 0.410). ISSN: 0891-773.
5. Tordecilla, R.; **Martins, L. C.**; Saiz, M.; Copado-Méndez, P. J.; Panadero, J.; Juan, A. A. (2021): [Agile Computational Intelligence for supporting Hospital Logistics during the COVID-19 Crisis](#). *Computational Management* (indexed in ISI SCI, 2020 IF = 4.193, Q2; 2020 SJR = 1.032, Q1). ISSN: 0969-6016.

10.3.2.2 Under Peer-Review

1. Tordecilla, R.; **Martins, L. C.**; Saiz, M.; Copado-Méndez, P. J.; Panadero, J.; Juan, A. A. (**u.r.**): Agile Computational Intelligence for supporting Hospital Logistics during the COVID-19 Crisis. *Modeling and Optimization in Science and Technologies* (indexed in Scopus, 2019 SJR = 0.204, Q3). ISSN: 2196-7326.
2. **Martins, L. C.**; Castaneda, J.; Juan, A. A.; Barrios, B.; Tondar, A.; Calvet, L.; Sanchez-Garcia, J. (**u.r.**): Supporting Efficient Assignment of Medical Resources in Cancer Treatments with Simulation-optimization. *Proceedings – Winter Simulation Conference* (Indexed in ISI WOS and Scopus, 2019 SJR = 0.410). ISSN: 0891-7736.
3. **Martins, L. C.**; Torres, M.; Perez-Bernabeu, E.; Corlu, C.; Juan, A. A.; Faulin, J. (**u.r.**): Solving a Urban Ridesharing Problem with Stochastic Traveling Times: a Simheuristic Approach. *Proceedings – Winter Simulation Conference* (Indexed in ISI WOS and Scopus, 2019 SJR = 0.410). ISSN: 0891-7736.
4. Tordecilla, R.; **Martins, L. C.**; Rabe, M.; Chicaiza-Vaca, J.; Juan, A. A. (**u.r.**): A Stochastic Team Orienteering Problem for a Hospital Logistics Case during the Covid-19 Crisis. *Proceedings – Winter Simulation Conference* (Indexed in ISI WOS and Scopus, 2019 SJR = 0.410). ISSN: 0891-7736.
5. Cardona, J.; Castaneda, J.; **Martins, L. C.**; Gandouz, M.; Franco, G.; Juan, A. A. (**u.r.**): Using Data Analytics & Machine Learning to Design Business Interruption Insurance Products for Rail Freight Operators. *Transportation Research Procedia*, 47:83-90 (indexed in Scopus, 2018 SJR = 0.601). ISSN: 2352-1465.
6. Castaneda, J.; Cardona, J.; **Martins, L. C.**; Juan, A. A. (**u.r.**): Supervised Machine Learning Algorithms for Measuring and Promoting Sustainable Transportation and Green Logistics. *Transportation Research Procedia*, 47:83-90 (indexed in Scopus, 2018 SJR = 0.601). ISSN: 2352-1465.

-
7. Tordecilla, R.; Copado-Méndez, P. J.; Panadero, J.; **Martins, L. C.**; Juan, A. A. ([u.r.](#)): An Agile and Reactive Biased-Randomized Heuristic for an Agri-Food Rich Vehicle Routing Problem. *Transportation Research Procedia*, 47:83-90 (indexed in Scopus, 2018 SJR = 0.601). ISSN: 2352-1465.

Appendices

A.1 Complete Production

In the following, the cover page of the papers developed during the development of this thesis is presented in chronological publication date. Firstly, the articles currently published in Journals and Conferences are presented, followed by submissions that, to date, are under peer-reviewing process.

A.1.1 ISI-JCR Indexed Journal Papers

A.1.1.1 Published

WILEY

INTERNATIONAL
TRANSACTIONS
IN OPERATIONAL
RESEARCHIntl. Trans. in Op. Res. 28 (2021) 201–221
DOI: 10.1111/itor.12796

Agile optimization of a two-echelon vehicle routing problem with pickup and delivery

Leandro do C. Martins^{a,*} , Patrick Hirsch^b  and Angel A. Juan^{a,c} ^a*IN3 – Computer Science Department, Universitat Oberta de Catalunya, Barcelona, Spain*^b*Institute of Production and Logistics, University of Natural Resources and Life Sciences, Vienna, Austria*^c*Euncet Business School, Terrassa, Spain**E-mail: leandrocm@uoc.edu [do C. Martins]; patrick.hirsch@boku.ac.at [Hirsch]; ajuanp@uoc.edu [Juan]*

Received 28 May 2019; received in revised form 8 January 2020; accepted 10 March 2020

Abstract

In this paper, we consider a vehicle routing problem in which a fleet of homogeneous vehicles, initially located at a depot, has to satisfy customers' demands in a two-echelon network: first, the vehicles have to visit intermediate nodes (e.g., a retail center or a consolidation center), where they deliver raw materials or bulk products and collect a number of processed items requested by the customers in their route; then, the vehicles proceed to complete their assigned routes, thus delivering the processed items to the final customers before returning to the depot. During this stage, vehicles might visit other intermediate nodes for reloading new items. In some real-life scenarios, this problem needs to be solved in just a few seconds or even milliseconds, which leads to the concept of “agile optimization.” This might be the case in some rescue operations using drones in humanitarian logistics, where every second can be decisive to save lives. In order to deal with this real-time two-echelon vehicle routing problem with pickup and delivery, an original constructive heuristic is proposed. This heuristic is able to provide a feasible and reasonably good solution in just a few milliseconds. The constructive heuristic is extended into a biased-randomized algorithm using a skewed probability distribution to modify its greedy behavior. This way, parallel runs of the algorithm are able to generate even better results without violating the real-time constraint. Results show that the proposed methodology generates competitive results in milliseconds, being able to outperform other heuristics from the literature.

Keywords: agile optimization; disaster management; two-echelon vehicle routing problem; biased-randomized algorithms

1. Introduction

Real-time optimization, where decisions need to be made in just a few seconds or even milliseconds, has many application areas in logistics. For example, in the event of disasters, real-time optimization

*Corresponding author.

© 2020 The Authors.

International Transactions in Operational Research © 2020 International Federation of Operational Research Societies
Published by John Wiley & Sons Ltd, 9600 Garsington Road, Oxford OX4 2DQ, UK and 350 Main St, Malden, MA02148, USA.



A biased-randomized iterated local search for the vehicle routing problem with optional backhauls

Julio C. Londoño¹ · Rafael D. Tordecilla^{2,3} · Leandro do C. Martins² · Angel A. Juan²

Received: 29 July 2019 / Accepted: 18 April 2020
© Sociedad de Estadística e Investigación Operativa 2020

Abstract

The vehicle routing problem with backhauls integrates decisions on product delivery with decisions on the collection of returnable items. In this paper, we analyze a scenario in which collection of items is optional—but subject to a penalty cost. Both transportation costs and penalties associated with non-collecting decisions are considered. A mixed-integer linear model is proposed and solved for small instances. Also, a metaheuristic algorithm combining biased randomization techniques with iterated local search is introduced for larger instances. Our approach yields cost savings and is competitive when compared to other state-of-the-art approaches.

Keywords Vehicle routing problem with optional backhauls · Returnable transport items · Biased randomization · Iterated local search

Mathematics Subject Classification 90B06 · 90C11 · 90C59

✉ Rafael D. Tordecilla
rtordecilla@uoc.edu

Julio C. Londoño
julio.londono@correounivalle.edu.co

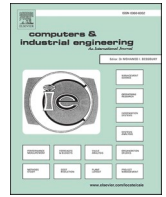
Leandro do C. Martins
leandrocm@uoc.edu

Angel A. Juan
ajuamp@uoc.edu

¹ Faculty of Engineering, Universidad del Valle, Cali, Colombia

² IN3-Computer Science Department, Universitat Oberta de Catalunya, 08860 Castelldefels, Spain

³ Faculty of Engineering, Universidad de La Sabana, Km 7 Autopista norte de Bogota, D.C., Chia, Colombia



A two-phase local search with a discrete-event heuristic for the omnichannel vehicle routing problem

Christopher Bayliss, Leandro do C. Martins^{*}, Angel A. Juan

IN3 – Computer Science Dept., Universitat Oberta de Catalunya, 08860 Castelldefels, Spain

ARTICLE INFO

Keywords:

Vehicle routing problem
Omnichannel distribution
Pick up and delivery
Discrete-event heuristics
Integer programming

ABSTRACT

This work tackles a pick up and delivery vehicle routing problem that emerges from the integration of delivery problems from two neighbouring layers of a supply chain: (i) retailer replenishment; and (ii) online customers deliveries, where online customer orders have to be picked up from retailers on route. This integration is motivated by recent developments in marketing, e.g.: the move to the omnichannel marketing paradigm, in which companies manage multiple sales channels in a seamless, unified, and integrated way. This paper proposes a novel model and an original solving approach. In order to tackle the complex capacity constraints, previous models propose to solve a delivery vehicle assignment problem before solving the associated vehicle routing problem. In contrast, our model deals with the whole routing problem by considering only the capacity feasible routing decisions. Our solving approach is based on a two-phase algorithm. In the first phase, a discrete-event constructive heuristic is employed. In the second phase, the most ‘promising’ solutions obtained in the previous phase are refined using a sequence of local search neighbourhoods. A series of extensive computational experiments show that our algorithm is able to identify new best-known solutions for the vast majority of problem instances. Also, for a set of small instances we obtain improved lower bounds compared to a previously proposed lower bound formulation.

1. Introduction

With recent advances in information technologies, consumers behavior has been changed, and classical problems in supply chain management have incorporated new constraints, decisions, and assumptions. The emerging online marketing channel has made possible the expansion of e-commerce. As a consequence, an abundance of information, opinions, and access to a vast combined supply of stock is available to consumers from all around the world. Today, the online channel represents a competitive marketing channel that has transformed e-commerce into a global trend as well as into an essential tool for business worldwide (Pagès-Bernaus, Ramalhinho, Juan, & Calvet, 2019). Omnichannel retailing refers to the integrated use of multiple buying and delivery channels in order to fulfill customer demands and to provide them with a seamless experience (Chopra, 2016). When immersed in an omnichannel experience, customers have access to different shopping channels. For instance, customers can purchase products by physically visiting a store or via online shopping. Hence, the orders can be picked up by the customers in physical facilities (e.g.,

retail centers or stores) or be delivered at home by using the delivery service provided by the retailers. Although this channel integration provides consumers with a seamless shopping experience (Zhang, Ren, Wang, & He, 2018), it also requires the design of integrated distribution systems. For instance, the same fleet of vehicles should be able to replenish the stocks of retail stores and, at the same time, service the product demands placed by online customers. Such integration of tasks introduces complex precedence constraints into vehicle route planning, since: (i) customer orders need to be picked up from retail centers before they can be delivered; and (ii) those retail centers need to be replenished simultaneously. According to Hübner, Holzapfel, and Kuhn (2016), the need for improving the delivery system in this retailing environment includes the development and optimization of delivery modes and the increase of the delivery speed –since most customers want to receive the delivery up to a maximum of two days after placing the order.

This paper focuses on the emerging omnichannel vehicle routing problem (OVRP), which is a rich extension of the classical vehicle routing problem (VRP) (Caceres-Cruz, Arias, Guimarans, Riera, & Juan, 2015). Some authors explore multi-depot scenarios considering factors

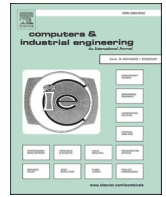
^{*} Corresponding author.

E-mail addresses: cbayliss@uoc.edu (C. Bayliss), leandrocm@uoc.edu (L.C. Martins), ajuanp@uoc.edu (A.A. Juan).



Contents lists available at ScienceDirect

Computers & Industrial Engineering

journal homepage: www.elsevier.com/locate/caie

Optimizing ride-sharing operations in smart sustainable cities: Challenges and the need for agile algorithms

Leandro do C. Martins^{a,*}, Rocio de la Torre^b, Canan G. Corlu^c, Angel A. Juan^a, Mohamed A. Masmoudi^a

^a IN3 - Computer Science Dept., Universitat Oberta de Catalunya, 08860 Castelldefels, Spain

^b INARBE Institute, Public University of Navarre, Pamplona, Spain

^c Metropolitan College, Boston University, Boston, USA

ARTICLE INFO

Keywords:

Ride-sharing optimization
Carpooling
Simulation
Metaheuristics
Real-time optimization
Smart sustainable cities

ABSTRACT

Mobility solutions like ride-sharing and carpooling are becoming popular in many urban and metropolitan areas around the globe. These solutions, however, create many operational challenges that need to be solved in order to make them more efficient and sustainable in time, e.g.: determining the number and location of parking slots, finding the optimal routes in terms of time or emissions, or developing synchronized schedules among ride-sharing users. This paper provides an updated review on car-sharing optimization studies (including ride-sharing and carpooling), compares different analytical approaches in this research area, and discusses the emerging concept of 'agile' algorithms as one of the approaches that might contribute to deal with the requirements of large-scale and dynamic car-sharing optimization problems.

1. Introduction

Transport and logistics (T&L) activities represent a key sector in modern societies, and they significantly contribute to their social and economic progress. At the same time, the raise of the on-demand economy (services) and the e-commerce activity (products) has boosted the number of pick-ups and deliveries in urban, metropolitan, and peri-urban areas. Thus, there is a need for increasing the effectiveness and sustainability of T&L activities and policies (Cui et al., 2020). Due to the increasing number of people who live in urban areas, many local and regional governments realize that T&L activities will play a major role in the development of the so-called smart sustainable cities (Bibri & Krogstie, 2019). Large quantities of data are gathered in real-time via electronic devices located inside vehicles and infrastructures (computer chips, sensors, traffic cameras, drones, etc.), transmitted over the Internet, and analyzed through information and expert systems (Mehmood et al., 2017). Monetary, environmental, and social costs associated with single occupancy vehicles could be reduced by more efficient utilization of empty seats in personal transportation vehicles. This is the goal of carpooling and ride-sharing strategies, which, apart from generating substantial economic impact to users, aim at reducing the

number of vehicles on the road and, as a consequence, contribute to diminishing traffic and pollution (Bistaffa, Blum, Cerquides, Farinelli, & Rodríguez-Aguilar, 2019). According to Schrank, Eisele, and Lomax (2019), the annual cost of congestion in the United States (U.S.) achieved \$ 166 billion in 2017, which caused Americans to lose around 8.8 billion hours on sitting in traffic and purchase an extra 3.3 billion gallons of fuel. Environmentally speaking, transportation counted for about 28% of total carbon dioxide-equivalent emissions (CO_2e) in the U.S. in 2018, being light-duty vehicles responsible for 59% of them (United States Environmental Protection Agency, 2020). In Europe, on the other hand, transportation was responsible for almost 30% of CO_2e in 2016, of which 72% comes from road transportation. Particularly, cars are responsible for almost 61% of these 72% of gas emissions (European Environment Agency, 2019). In an effort to minimize related problems, such as greenhouse effects and global warming, the European Union developed a strategic plan for low-emission mobility. As stated in European Commission (2016), one of the main elements on which this strategy relies on refers to increasing the efficiency of the transport system by benefiting from digital technologies, smart pricing, and further encouraging the shift to lower emission sustainable transportation modes. Therefore, the need for smarter and sustainable

* Corresponding author.

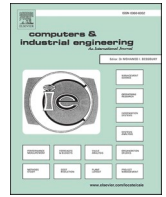
E-mail addresses: leandrocm@uoc.edu (L.C. Martins), rocio.delatorre@unavarra.es (R. de la Torre), canan@bu.edu (C.G. Corlu), ajuanp@uoc.edu (A.A. Juan), mmasmoudim@uoc.edu (M.A. Masmoudi).

<https://doi.org/10.1016/j.cie.2020.107080>

Received 21 March 2020; Received in revised form 4 November 2020; Accepted 19 December 2020

Available online 23 December 2020

0360-8352/© 2020 Elsevier Ltd. All rights reserved.



Combining production and distribution in supply chains: The hybrid flow-shop vehicle routing problem

Leandro do C. Martins^{a,*}, Eliana M. Gonzalez-Neira^{b,c}, Sara Hatami^a, Angel A. Juan^a, Jairo R. Montoya-Torres^c

^a IN3 – Computer Science Department, Universitat Oberta de Catalunya, Barcelona, Spain

^b Departamento de Ingeniería Industrial, Pontificia Universidad Javeriana, Bogotá, Colombia

^c Grupo de Investigación en Sistemas Logísticos, Universidad de La Sabana, Chía, Colombia

ARTICLE INFO

Keywords:

Hybrid flow-shop problem
Vehicle routing problem
Biased randomization
Metaheuristics

ABSTRACT

Many supply chains are composed of producers, suppliers, carriers, and customers. These agents must be coordinated to reduce waste and lead times. Production and distribution are two essential phases in most supply chains. Hence, improving the coordination of these phases is critical. This paper studies a combined hybrid flow-shop and vehicle routing problem. The production phase is modeled as a hybrid flow-shop configuration. In the second phase, the produced jobs have to be delivered to a set of customers. The delivery is carried out in batches of products, using vehicles with a limited capacity. With the objective of minimizing the service time of the last customer, we propose a biased-randomized variable neighborhood descent algorithm. Different test factors, such as the use of alternative initial solutions, solution representations, and loading strategies, are considered and analyzed.

1. Introduction

In most supply chains, there is an increasing need to coordinate the efforts of suppliers, producers, and carriers to efficiently deliver products to customers, so that waste and lead times are reduced. The production and distributions phases are critical in any supply chain: finished products are transferred from production centers to a warehouse or distribution centers by cargo vehicles. In order to enhance the operational performance, both phases need to be considered while optimizing operations. Still, due to the complexity of these phases, traditional approaches usually consider them as two isolated problems (Chen, 2010).

In this paper, we offer a more holistic approach by considering the production and distribution phases altogether. This is the case, for example, of distributing medical tests or vaccines to local health centers –so they can be administrated to the population as soon as possible– while these items are being produced, in large quantities, at a central laboratory. Hence, the production phase is modeled as a hybrid flow-shop (HFS) environment, while the distribution phase is modeled as a vehicle routing problem (VRP). Accordingly, the combined problem can

be referred to as a hybrid flow-shop vehicle routing problem (HFS-VRP). As shown in Fig. 1, in the production phase a set J of jobs (items) are processed. Each job has to go through a set S of sequential stages. At each stage $s \in S$, a set M_s of parallel and identical machines are available to process the job. Given a job $j \in J$, its processing time in stage $s \in S$ is given by $p_{js} > 0$. Regarding the distribution phase, a set C of customers and a single vehicle that makes multiple trips are considered. In each trip the vehicle delivers a batch of jobs. Each job $j \in J$ allows to a specific customer $c \in C$ and occupies a volume of $q_j > 0$, being $Q > \max_{i \in J} \{q_i\}$ the maximum loading capacity of the vehicle.

In order to speed up the delivery process, finished items are grouped into batches that can be delivered to customers while the production system is manufacturing new ones. In this context, the goal is to minimize the total time elapsed since the start of the manufacturing process and the delivery of the last customer's demand, i.e., the makespan of the hybrid problem. In order to solve the proposed HFS-VRP, three different and interrelated decisions have to be made: (i) determining the job sequence on each machine at the production phase; (ii) assigning the finished jobs to a proper batch for deliver; and (iii) determining adequate route for each trip of the vehicle in order to deliver jobs to customers.

* Corresponding author.

E-mail addresses: leandrocm@uoc.edu (L.C. Martins), eliana.gonzalez@javeriana.edu.co (E.M. Gonzalez-Neira), shatami@uoc.edu (S. Hatami), ajuanp@uoc.edu (A.A. Juan), jairo.montoya@unisabana.edu.co (J.R. Montoya-Torres).

<https://doi.org/10.1016/j.cie.2021.107486>

Received 29 September 2020; Received in revised form 30 April 2021; Accepted 14 June 2021

Available online 18 June 2021

0360-8352/© 2021 Elsevier Ltd. All rights reserved.

Agile optimization for a real-time facility location problem in Internet of Vehicles networks

Leandro do C. Martins¹ | Daniele Tarchi² | Angel A. Juan¹ | Alessandro Fusco²

¹IN3, Computer Science Dept., Universitat Oberta de Catalunya, Barcelona, Spain

²Department of Electrical, Electronic and Information Engineering "Guglielmo Marconi", University of Bologna, Bologna, Italy

Correspondence

Leandro do C. Martins, IN3, Universitat Oberta de Catalunya, 08018 Barcelona, Spain.
Email: leandrocm@uoc.edu

Abstract

The uncapacitated facility location problem (UFLP) is a popular *NP-hard* optimization problem that has been traditionally applied to logistics and supply networks, where decisions are difficult to reverse. However, over the years, many new application domains have emerged, in which real-time optimization is needed, such as Internet of Vehicles (IoV), virtual network functions placement, and network controller placement. IoV scenarios take into account the presence of multiple roadside units (RSUs) that should be frequently assigned to operating vehicles. To ensure the desired quality of service level, the allocation process needs to be carried out frequently and efficiently, as vehicles' demands change. In this dynamic environment, the mapping of vehicles to RSUs needs to be reoptimized periodically over time. Thus, this article proposes an *agile optimization* algorithm, which is tested using existing benchmark instances. The experiments show that it can efficiently generate high-quality and real-time results in dynamic IoV scenarios.

KEYWORDS

agile optimization, biased-randomized heuristics, Internet of Vehicles, real-time optimization, smart cities, uncapacitated facility location problem

1 | INTRODUCTION

The uncapacitated facility location problem (UFLP) [4], is a well-known *NP-hard* optimization problem in the operations research literature. When applied in the logistics field, it typically refers to strategic decisions in a static environment. Hence, the focus is not on providing "good" solutions fast but on providing near-optimal ones regardless of the computational times. However, over the years many new application domains of the UFLP emerged, and some of them require the computer to frequently generate new customers-to-facilities mappings as customers' demands change over time. In this dynamic context, computing time becomes a relevant factor, since it affects our capacity to make decisions as these are required [8]. Accordingly, there is a need for "agile" solution methodologies that can quickly provide high-quality solutions in dynamic environments. Examples of such environments can be found in the health-care sector, where demand points, operating and distribution costs can vary over time, and new facilities might be added at different time periods [1, 37, 44]. Telecommunication systems constitute one of these application areas where computing times are crucial. The 5G communication standard has defined several usage cases and applications where low latency communications need to be addressed [40]. Examples of such applications are the Internet of Vehicles (IoV) [35], the network functions virtualization (NFV) [31], and the controller placement problem (CPP) [41], in which decisions on *how many* facilities are needed and *where* to place them might influence the Quality of Service (QoS) level offered to the users.

The IoV can be defined as a distributed transport network that is capable of making its own decisions about driving customers to their destinations. This is achieved by having communications, storage, intelligence, and learning capabilities that allow the system to predict customers' intentions [26]. In order to properly work, the IoV network should be based on a telecommunication infrastructure and able to connect every vehicle and node in the system. The vehicular communication infrastructure has

Article

Electric Vehicle Routing, Arc Routing, and Team Orienteering Problems in Sustainable Transportation

Leandro do C. Martins ¹, Rafael D. Tordecilla ^{1,2}, Juliana Castaneda ¹, Angel A. Juan ^{1,*} and Javier Faulin ³

¹ IN3–Computer Science Department, Universitat Oberta de Catalunya, 08018 Barcelona, Spain; leandrocm@uoc.edu (L.d.C.M.); rtordecilla@uoc.edu (R.D.T.); jcastanedaji@uoc.edu (J.C.)

² School of Engineering, Universidad de La Sabana, Chia 250001, Colombia

³ Institute of Smart Cities, Dept. Statistics, Computer Sciences, and Mathematics, Public University of Navarre, 31006 Pamplona, Spain; javier.faulin@unavarra.es

* Correspondence: ajuanp@uoc.edu

Abstract: The increasing use of electric vehicles in road and air transportation, especially in last-mile delivery and city mobility, raises new operational challenges due to the limited capacity of electric batteries. These limitations impose additional driving range constraints when optimizing the distribution and mobility plans. During the last years, several researchers from the Computer Science, Artificial Intelligence, and Operations Research communities have been developing optimization, simulation, and machine learning approaches that aim at generating efficient and sustainable routing plans for hybrid fleets, including both electric and internal combustion engine vehicles. After contextualizing the relevance of electric vehicles in promoting sustainable transportation practices, this paper reviews the existing work in the field of electric vehicle routing problems. In particular, we focus on articles related to the well-known vehicle routing, arc routing, and team orienteering problems. The review is followed by numerical examples that illustrate the gains that can be obtained by employing optimization methods in the aforementioned field. Finally, several research opportunities are highlighted.

Keywords: electric batteries; vehicle routing problem; arc routing problem; team orienteering problem



Citation: Martins, L.C.; Tordecilla, R.D.; Castaneda, J.; Juan, A.A.; Faulin, J. Electric Vehicle Routing, Arc Routing, and Team Orienteering Problems in Sustainable Transportation. *Energies* **2021**, *14*, 5131. <https://doi.org/10.3390/en14165131>

Academic Editors: Daniel J. Auger and Jorge Barreras

Received: 18 July 2021

Accepted: 17 August 2021

Published: 19 August 2021

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).







1. Introduction

With the goal of promoting sustainability, many cities in the world are observing an increasing use of electric vehicles (EVs), both for citizens' mobility [1] and for last-mile logistics [2]. The use of zero-emission technologies is supported by governmental plans in regions such as Europe [3], North America [4], and Asia [5]. According to Kapustin and Grushevenko [6], EVs will account for a noticeable share (between 11% and 28%) of the road transportation fleet by 2040. Still, many authors point out batteries' driving range anxiety, high recharging times, scarcity of recharging stations, and lack of effective financial incentives that compensate for the higher cost of most EV models as some of the main barriers for the generalization of EVs in our cities [7–9].

In urban, peri-urban, and metropolitan areas, many activities related to freight transportation and citizens' mobility are carried out by fleets of vehicles [10]. The efficient coordination of these fleets becomes necessary in order to reduce monetary costs, operation times, energy consumption, and environmental/social impacts on the city. However, this coordination constitutes a relevant challenge that is typically modeled as a mathematical optimization problem. Depending on the specific characteristics of the transportation activity, different families of problems can be found in the scientific literature. Among the most popular ones, we can include vehicle routing problems (VRPs) [11–13], arc routing problems (ARPs) [14,15], and team orienteering problems (TOPs) [16,17]. These problems, which can model scenarios involving both road and aerial EVs, are NP-hard even in their

Article

Fuzzy Simheuristics for Optimizing Transportation Systems: Dealing with Stochastic and Fuzzy Uncertainty

Rafael D. Tordecilla ^{1,2}, Leandro do C. Martins ¹, Javier Panadero ^{1,3}, Pedro J. Copado ^{1,3},
Elena Perez-Bernabeu ⁴ and Angel A. Juan ^{1,3,*}

¹ IN3–Computer Science Department, Universitat Oberta de Catalunya, 08018 Barcelona, Spain; rtordecilla@uoc.edu (R.D.T.); leandrocm@uoc.edu (L.d.C.M.); jpanaderom@uoc.edu (J.P.); pcpadom@uoc.edu (P.J.C.)

² School of Engineering, Universidad de La Sabana, Chia 250001, Colombia

³ Euncet Business School, 08225 Terrassa, Spain

⁴ Department of Applied Statistics and Operations Research, Universitat Politècnica de València, 03801 Alcoy, Spain; elenapb@eio.upv.es

* Correspondence: ajuanp@uoc.edu

Abstract: In the context of logistics and transportation, this paper discusses how simheuristics can be extended by adding a fuzzy layer that allows us to deal with complex optimization problems with both stochastic and fuzzy uncertainty. This hybrid approach combines simulation, metaheuristics, and fuzzy logic to generate near-optimal solutions to large scale NP-hard problems that typically arise in many transportation activities, including the vehicle routing problem, the arc routing problem, or the team orienteering problem. The methodology allows us to model different components—such as travel times, service times, or customers’ demands—as deterministic, stochastic, or fuzzy. A series of computational experiments contribute to validate our hybrid approach, which can also be extended to other optimization problems in areas such as manufacturing and production, smart cities, telecommunication networks, etc.

Keywords: transportation; vehicle routing problems; metaheuristics; simulation-optimization; fuzzy techniques



Citation: Tordecilla, R.D.; Martins, L.d.C.; Panadero, J.; Copado, P.J.; Perez-Bernabeu, E.; Juan, A.A. Fuzzy Simheuristics for Optimizing Transportation Systems: Dealing with Stochastic and Fuzzy Uncertainty. *Appl. Sci.* **2021**, *11*, 7950. <https://doi.org/10.3390/app11177950>

Academic Editor: Ludmila Dymova

Received: 24 July 2021

Accepted: 26 August 2021

Published: 28 August 2021

Publisher’s Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Copyright: © 2021 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

Managers tend to rely on analytical methods that allow them to make informed decisions. This explains why optimization models play a key role in many industries and business, including the logistics and transportation sector. Whenever accurate information on the inputs and constraints of the optimization problem is available, the resulting deterministic models can be solved by using well-known methods, either of exact or approximate nature.

Many optimization problems in real-life transportation involve taking into account a large number of variables and rich constraints, which often makes them to be NP-hard [1]. When this is the case, the computational complexity makes it difficult to obtain optimal solutions in a short computational time. At this point, heuristic approaches can provide near-optimal solutions that, in turn, cover all the requirements of the problem [2]. When dealing with challenging optimization problems, there is a tendency to divide them into sub-problems, which simplifies the difficulty but might also lead to sub-optimal solutions [3,4]. Given the increase in computational power experienced during the last decade, and also the development of advanced metaheuristic algorithms, it is possible nowadays to solve rich and large-scale problems that were intractable in the past [5].

In the scientific literature on combinatorial optimization problems, it is often assumed that the input values are constant and known. However, in a real-world scenario this is rarely the case, since uncertainty is often present and affects these inputs. In the context

A.1.1.2 Under Review

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier 10.1109/ACCESS.2021.DOI

An Agile Optimization Approach for the Multi-Period Internet of Vehicles Problem

LUCA CESARANO¹, ANDREA CROCE¹, LEANDRO DO C. MARTINS², DANIELE TARCHI¹ (Senior Member, IEEE), AND ANGEL A. JUAN²

¹University of Bologna, Viale del Risorgimento 2, Bologna, 40136, Italy (email: luca.cesarano@studio.unibo.it, andrea.croce3@studio.unibo.it, daniele.tarchi@unibo.it)

²IN3 – Computer Science Dept., Universitat Oberta de Catalunya, Barcelona, Spain (email: leandrocm@uoc.edu, ajuanp@uoc.edu)

Corresponding author: Daniele Tarchi (e-mail: daniele.tarchi@unibo.it).

This work has been done during the abroad study period of L. Cesarano and A. Croce at Universitat Oberta de Catalunya, Spain, supported by Erasmus+ Study programme of the European Union.

ABSTRACT Emerging technologies, such as self-driving vehicles and 5G communications, are raising new mobility and transportation possibilities in smart and sustainable cities, bringing to a new eco-system often referred to as Internet of Vehicle (IoV). However, IoV also requires efficient algorithms that can support real-time information sharing and vehicle routing. This paper analyzes the Multi-Period Internet of Vehicles Problem (MPIoVP), where services running on multiple roadside units (RSUs) need to be frequently assigned and re-assigned to operating self-driving vehicles. The allocation processes need to be done frequently, quickly, and efficiently, as vehicles move around the city. In order to solve this multi-period version of the IoVP, an agile algorithm is proposed. The algorithm makes use of biased-randomization techniques for agile optimization. The resulting methodology is tested against a set of benchmark instances in order to illustrate its potential.

INDEX TERMS Internet of Vehicle, Smart Cities, Uncapacitated Facility Location Problem, Real-time Optimization, Energy Minimization.

I. INTRODUCTION

EACH form of innovation aiming to achieve sustainability and to optimize the use of resources in responsible ways is defined as an eco-innovation form. The need for developing green-growth IT strategies in urban environments is raising day by day, and has the noble aim of improving citizens' quality of life, apart from ensuring that future generations will be able to meet basic and first necessity needs – e.g., clean air, water, and biodiversity– thanks to adjustments like the reduction in traffic and in emission of greenhouse gases [1].

Modern technologies, like 5G and Internet of Things (IoT), can be employed to support the development of *green cities* [2]–[4]: a huge distributed cloud infrastructure made by thousands of smart devices (typically small and embedded ones), which are able to communicate with each other in real-time [5]. Among several application domains, the Internet on Vehicles (IoV) [6], a sub-domain of IoT, can be described as a distributed and interconnected network that supports the need for sharing data created by vehicles –but could include also pedestrians, bikes, and other urban objects– in a real-time fashion with roadside units (RSU). These are special facilities

that act as edge computing nodes, running different services and able to jointly convey communications and process data, without requiring transfer to central cloud facilities [7].

In such a complex scenario (FIGURE 1), one of the main challenges to be faced when solving an IoV problem is to optimally place the RSUs so that an optimal service coverage of vehicles is provided with respect to the some target quality of service (QoS) parameters –e.g., service coverage, throughput, low latency, and energy consumption [8]–[10]. In addition, roads are very dynamic environments: cities may be congested in some areas at specific times of a day, with vehicles moving at a different speed. This dynamism of the environment must be taken into account as well.

The Multi-Period Internet on Vehicles Problem (MPIoVP) can be seen as a rich variant of the IoVP, in which the RSUs configuration has to react and quickly adapt to the ever-changing traffic and connection requirements from the vehicles, while keeping the energy consumption low. Thus, every few minutes, a near-optimal configuration of RSUs might need to be re-computed in a short amount of time (typically within a few seconds). In FIGURE 2, the same city blocks are depicted in three different successive periods,

Intl. Trans. in Op. Res. xx (2021) 1–26

A Learnheuristic Algorithm for the Temporary-Facility Location and Queuing Problem during Epidemic Events

Christopher Bayliss^{a,*}, Angel A Juan^b, Javier Panadero^b, Pedro J. Copado^b and Leandro do C. Martins^b

^aManagement School, University of Liverpool, Liverpool L69 7ZH, UK

^bIN3 – Computer Science Dept., Universitat Oberta de Catalunya,

Av. Carl Friedrich Gauss, 5, Castelldefels, Barcelona, 08860, Spain

E-mail: Christopher.Bayliss@liverpool.ac.uk [C. Bayliss]; ajuanp@uoc.edu [A. Juan];

jpanaderom@uoc.edu [J. Panadero];

pcopadom@uoc.edu [P. Copado];

leandrocm@uoc.edu [L. Martins]

Received DD MMMM YYYY; received in revised form DD MMMM YYYY; accepted DD MMMM YYYY

Abstract

Epidemic outbreaks, such as the one generated by the coronavirus disease, have raised the need for more efficient healthcare logistics. One of the challenges that many governments have to face in such scenarios is the deployment of temporary medical facilities across a region with the purpose of providing medical services to their citizens. This work tackles this temporary-facility location and queuing problem with the goals of minimising costs, the expected completion time, population travel and waiting times. The completion time for a facility depends on the numbers assigned to those facilities as well as stochastic arrival times. This work proposes a learnheuristic algorithm to solve the facility location and population assignment problem. Firstly a machine learning algorithm is trained using data from a queuing model (simulation module). The learnheuristic then constructs solutions using the machine learning algorithm to rapidly evaluate decisions in terms of facility completion and population waiting times. The efficiency and quality of the algorithm is demonstrated by comparison with exact and simulation-only methodologies. A series of experiments are performed which explore the trade offs between solution cost, completion time, population travel and waiting times.

Keywords: Facilities planning and design; Queuing; Metaheuristics; Simulation, Machine learning.

1. Introduction

Logistics plays a vital role in a large number of contexts. Apart from being massively a tool of study in supply chain systems in the last decades, it has emerged in substantially different fields, such as

* Author to whom all correspondence should be addressed (e-mail: Christopher.Bayliss@liverpool.ac.uk).

Edge Computing and IoT Analytics for Agile Optimization in Intelligent Transportation Systems

Mohammad Peyman^{1*} , Pedro J. Copado^{1,2} , Rafael D. Tordecilla^{1,3} , Leandro do C. Martins¹ , Fatos Xhafa⁴ 
and Angel A. Juan^{1,2} 

¹ IN3 – Computer Science Department, Universitat Oberta de Catalunya, 08018 Barcelona, Spain, {mpeyman, pcpadom, rtordecilla, leandrocm, ajuanp}@uoc.edu

² Euncet Business School, Barcelona 08018, Spain

³ School of Engineering, Universidad de La Sabana, Chia, Colombia

⁴ Computer Science Department, Universitat Politècnica de Catalunya, 08034 Barcelona, Spain, fatos@cs.upc.edu

* Correspondence: mpeyman@uoc.edu

Abstract: With the emergence of fog and edge computing, new possibilities arise regarding the data-driven management of citizens' mobility in smart cities. Internet of Things (IoT) Analytics refers to the use of these technologies, data, and analytical models to describe the current status of the city traffic, to predict its evolution over the next hours, and to make decisions that increase the efficiency of the transportation system. In this paper, we review the state of the art in this topic by highlighting related work over which we will build our integrated concept of IoT Analytics and 'agile' optimization algorithms. These algorithms allow us to process, in real time, the data gathered from IoT systems in order to optimize automatic decisions on the city transportation system, including: optimizing the vehicle routing, recommending customized transportation modes to the citizens, generating efficient ride-sharing and car-sharing strategies, etc. An illustrative example regarding the use of open data and agile algorithms is provided to illustrate the potential of this approach.

Keywords: Fog; Edge Computing; Internet of Things; Intelligent Transportation Systems; Smart Cities; Machine Learning; Agile Optimization

Citation: Edge Computing and IoT in ITS. *Energies* **2021**, *1*, 0. <https://doi.org/>

Received:

Accepted:

Published:

Publisher's Note: MDPI stays neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Copyright: © 2021 by the authors. Submitted to *Energies* for possible open access publication under the terms and conditions of the Creative Commons Attribution (CC BY) license (<https://creativecommons.org/licenses/by/4.0/>).

1. Introduction

In today's modern society, urban centers are facing the so-called booming of information. Due to the population growth in many countries around the globe, and recent innovations in information and telecommunication technologies, several activities and related challenges have jointly arisen. People are increasingly consuming more information through their mobile devices, vehicles are equipped with different intelligent systems, devices are distributed around the cities for gathering and generating information, and urban areas are continuously taking advantage of these information technologies and big data. Consequently, the so-called smart cities have emerged, whose scope combines sustainable development with the intelligent management of gathered data in order to enhance the operation of different services within urban areas, such as waste collection management [1], car-sharing / ride-sharing activities [2], the optimal location of recharging stations for electric vehicles (EVs), among others. In this matter, during the past few years, the Internet of things (IoT) has become a popular term that plays a significant role to expand and produce a lot of data through sensors and allows citizens and things to be connected in any situation or with anyone [3]. Also, fog and cloud computing come to support IoT to manage the large amount of generated data (Figure 1) [4].

One of the main tasks when building smart cities is the development of intelligent transportation systems (ITS). These systems need to establish exact, effective, compre-

A.1.2 Scopus Indexed Journal Papers

A.1.2.1 Published

Article

Using Biased-Randomized Algorithms for the Multi-Period Product Display Problem with Dynamic Attractiveness

Mage Marmol ¹, Leandro do C. Martins ², Sara Hatami ² and Angel A. Juan ^{1,2,*}
and Vicenc Fernandez ³

¹ Marketing Department, Euncet Business School, 08225 Terrassa, Spain; mage.marmol@euncet.es

² IN3—Computer Science Department, Universitat Oberta de Catalunya, 08018 Barcelona, Spain; leandroc@uoc.edu (L.d.C.M.); shatami@uoc.edu (S.H.)

³ TechTalent-Lab, Universitat Politecnica de Catalunya—BarcelonaTech, 08222 Terrassa, Spain; vicenc.fernandez@upc.edu

* Correspondence: ajuanp@uoc.edu; Tel.: +34-932-53-2300

Received: 15 October 2019; Accepted: 29 January 2020; Published: 1 February 2020



Abstract: From brick-and-mortar stores to omnichannel retail, the efficient selection of products to be displayed on store tables, advertising brochures, or online front pages has become a critical issue. One possible goal is to maximize the overall ‘attractiveness’ level of the displayed items, i.e., to enhance the shopping experience of our potential customers as a way to increase sales and revenue. With the goal of maximizing the total attractiveness value for the visiting customers over a multi-period time horizon, this paper studies how to configure an assortment of products to be included in limited display spaces, either physical or online. In order to define a realistic scenario, several constraints are considered for each period and display table: (i) the inclusion of both expensive and non-expensive products on the display tables; (ii) the diversification of product collections; and (iii) the achievement of a minimum profit margin. Moreover, the attractiveness level of each product is assumed to be dynamic, i.e., it is reduced if the product has been displayed in a previous period (loss of novelty) and vice versa. This generates dependencies across periods. Likewise, correlations across items are also considered to account for complementary or substitute products. In the case of brick-and-mortar stores, for instance, solving this rich multi-period product display problem enables them to provide an exciting experience to their customers. As a consequence, an increase in sales revenue should be expected. In order to deal with the underlying optimization problem, which contains a quadratic objective function in its simplest version and a non-smooth one in its complete version, two biased-randomized metaheuristic algorithms are proposed. A set of new instances has been generated to test our approach and compare its performance with that of non-linear solvers.

Keywords: omnichannel retail stores; product display problem; multi-period decisions; dynamic attractiveness; biased-randomized heuristics

1. Introduction

As discussed in Verhoef et al. [1], customers today are changing the way they decide where, how, and even when to buy. With the rise of Internet-based technologies and mobile devices, different shopping channels have appeared and attracted customers’ attention. Hence, e-commerce not only offers customers the possibility of browsing through different stores in an online environment, but also the ability to get information, opinions, and a vast availability of stock. Omnichannel commerce is a fully-integrated approach to e-commerce that provides customers with a unified experience

Article

A Simheuristic Algorithm for Solving the Stochastic Omnichannel Vehicle Routing Problem with Pick-up and Delivery

Leandro do C. Martins *, Christopher Bayliss, Pedro J. Copado-Méndez, Javier Panadero and Angel A. Juan

Internet Interdisciplinary Institute (IN3)–Computer Science Department, Universitat Oberta de Catalunya, 08018 Barcelona, Spain; cbayliss@uoc.edu (C.B.); pcopadom@uoc.edu (P.J.C.-M.); jpanaderom@uoc.edu (J.P.); ajuanp@uoc.edu (A.A.J.)

* Correspondence: leandrocm@uoc.edu

Received: 24 August 2020; Accepted: 15 September 2020; Published: 19 September 2020



Abstract: Advances in information and communication technologies have made possible the emergence of new shopping channels. The so-called ‘omnichannel’ retailing mode allows customers to shop for products online and receive them at home. This paper focuses on the omnichannel delivery concept for the retailing industry, which addresses the replenishment of a set of retail stores and the direct shipment of the products to customers within an integrated vehicle routing formulation. Due to its *NP-Hardness*, a constructive heuristic, which is extended into a biased-randomized heuristic and which is embedded into a multi-start procedure, is introduced for solving the large-sized instances of the problem. Next, the problem is enriched by considering a more realistic scenario in which travel times are modeled as random variables. For dealing with the stochastic version of the problem, a simheuristic algorithm is proposed. A series of computational experiments contribute to illustrate how our simheuristic can provide reliable and low-cost solutions under uncertain conditions.

Keywords: omnichannel retail stores; vehicle routing problem; pick-up and delivery; biased-randomized heuristics; simheuristics

1. Introduction

Today, people are changing their shopping behavior. Recent advances in information and communication technologies have introduced new shopping channels and models, which make possible the expansion of e-commerce and, consequently, the emergence of new challenges in operational research, transportation, and logistics areas. Specifically, regarding modern e-commerce business models, new decision variables and constraints have been incorporated in them, leading to emerging variants of distribution problems in supply chain management.

Unlike brick-and-mortar stores, where salespeople are available to support and help customers to make their purchases, the popularization of mobile devices with access to the Internet has promoted the use of different shopping channels. The online channel is an example that has emerged as a competitive marketing channel to that of traditional retail centers, transforming e-commerce into a global trend and an important tool for every business worldwide [1]. With e-commerce, customers are immersed in an environment of a plethora of information, opinions, and access to a vast combined supply of stock, which together allows them to browse through different stores in an online environment. According to some experts, the online shopping channels were predicted to kill off the physical ones. However, they co-exist and have completely transformed the way customers shop nowadays [2]. The use of a variety of shopping channels is referred to as ‘omnichannel retailing,’ where, instead of having only the single option of physically visiting a store to buy products, consumers can also buy them via online

Chapter 18

Agile Computational Intelligence for Supporting Hospital Logistics During the COVID-19 Crisis



Rafael D. Tordecilla, Leandro do C. Martins, Miguel Saiz, Pedro J. Copado-Mendez, Javier Panadero, and Angel A. Juan

Abstract This chapter describes a case study regarding the use of ‘agile’ computational intelligence for supporting logistics in Barcelona’s hospitals during the COVID-19 crisis in 2020. Due to the lack of sanitary protection equipment, hundreds of volunteers, the so-called “Coronavirus Makers” community, used their home 3D printers to produce sanitary components, such as face covers and masks, which protect doctors, nurses, patients, and other civil servants from the virus. However, an important challenge arose: how to organize the daily collection of these items from individual homes, so they could be transported to the assembling centers and, later, distributed to the different hospitals in the area. For over one month, we have designed daily routing plans to pick up the maximum number of items in a limited time—thus reducing the drivers’ exposure to the virus. Since the problem characteristics were different each day, a series of computational intelligence algorithms was employed. Most of them included flexible heuristic-based approaches and biased-randomized

R. D. Tordecilla (✉) · L. C. Martins · M. Saiz · P. J. Copado-Mendez · J. Panadero · A. A. Juan
IN3—Computer Science Department, Universitat Oberta de Catalunya, 08018 Barcelona, Spain
e-mail: rtordecilla@uoc.edu

L. C. Martins
e-mail: leandrocm@uoc.edu

M. Saiz
e-mail: msaiz@weoptimize.net

P. J. Copado-Mendez
e-mail: pcopadom@uoc.edu

J. Panadero
e-mail: jpanaderom@uoc.edu

A. A. Juan
e-mail: ajuanp@uoc.edu

R. D. Tordecilla
Universidad de La Sabana, Chía, Cundinamarca, Colombia

M. Saiz
weOptimize, 43700 El Vendrell, Spain

P. J. Copado-Mendez · J. Panadero · A. A. Juan
Euncet Business School, Terrassa 08225, Spain

A.1.3 Peer-reviewed Conference Papers

A.1.3.1 Published

22nd EURO Working Group on Transportation Meeting, EWGT 2019, 18-20 September 2019,
Barcelona, Spain

A Savings-Based Heuristic for Solving the Omnichannel Vehicle Routing Problem with Pick-up and Delivery

Leandro do C. Martins^{a,*}, Christopher Bayliss^a, Angel A. Juan^{a,b}, Javier Panadero^{a,b},
Mage Marmol^b

^aIN3 – Computer Science Dept., Universitat Oberta de Catalunya, Av. Carl Friedrich Gauss 5, 08860 Castelldefels, Spain

^bEuncet Business School, Ctra. Terrassa a Talamanca, 08225 Terrassa, Spain

Abstract

In recent times, new models of commerce have incorporated new decisions and constraints which have led to new variants of classical problems in supply chain management. Modern advances in Information and Communication Technologies have increased the number of marketing channels that are available to consumers. This paper focusses on the new “omnichannel” delivery concept for the retailing industry which addresses the replenishment of a set of retail stores and on the direct shipment of the products to customers (last-mile delivery) within an integrated VRP formulation. The VRP in omnichannel distribution systems consists of a group of retail stores that must be served from a distribution center and a set of online consumers that must be served by the same fleet of cargo vehicles from these retail stores. Since the VRP in omnichannel distribution systems is an NP-Hard problem, we propose a savings-based heuristic for solving large-size instances the VRP in omnichannel retailing. Results show that the proposed heuristic is able to find feasible and competitive solutions in a very short computational time.

© 2020 The Authors. Published by Elsevier B.V.

This is an open access article under the CC BY-NC-ND license (<http://creativecommons.org/licenses/by-nc-nd/4.0/>)

Peer-review under responsibility of the scientific committee of the 22nd Euro Working Group on Transportation Meeting

Keywords: Vehicle Routing Problem; Omnichannel Supply Chain; Savings Heuristic

1. Introduction

Today, new models of commerce have incorporated new decisions and constraints which have led to new variants of classical problems in supply chain management. In this context, the general Vehicle Routing Problem (VRP) is an operational decision which aims to design cargo vehicle routes with minimum transportation costs in order to distribute goods between depots and a set of consumers during a planning period without causing stockouts at any of the customers (Crainic and Laporte, 2012).

Traditionally, customers made their purchases in brick-and-mortar stores with the help of salespeople to find what they wanted or needed. However, with recent advances in Information and Communication Technologies (ICT) and

* Corresponding author.

E-mail address: leandrocm@uoc.edu

A SIMHEURISTIC-LEARNHEURISTIC ALGORITHM FOR THE STOCHASTIC TEAM ORIENTEERING PROBLEM WITH DYNAMIC REWARDS

Christopher Bayliss
Pedro J. Copado-Mendez
Javier Panadero
Angel A. Juan
Leandro do C. Martins

IN3 – Computer Science Department
Universitat Oberta de Catalunya
Av. Carl Friedrich Gauss, 5
Castelldefels, 08860, SPAIN

ABSTRACT

In this paper, we consider the stochastic team orienteering problem with dynamic rewards (STOPDR) and stochastic travel times. In the STOPDR, the goal is to generate routes for a fixed set of vehicles such that the sum of the rewards collected is maximized while ensuring that nodes are visited before a fixed time limit expires. The rewards associated with each node are dependent upon the times at which they are visited. Also, the dynamic reward values have to be learnt from simulation experiments during the search process. To solve this problem, we propose a biased-randomized learnheuristic (BRLH), which integrates a learning module and a simulation model. Randomization is important for generating a wide variety of solutions that capture the trade-off between reward and reliability. A series of computational experiments are carried out in order to analyze the performance of our BRLH approach.

1 INTRODUCTION

The team orienteering problem (TOP) was first studied by Chao et al. (1996). It consists of generating time-constrained routes through a graph, for a fixed-size fleet of m vehicles, such that the rewards collected from node visits is maximized. Each of the m vehicles' tours begins and ends at a depot node. The stochastic TOP (STOP) is an extension of the TOP where travel times between nodes or node rewards are uncertain. In this work, only the rewards collected within a limited amount of time count. The STOP introduces the need to consider not only reward maximization, but also solution reliability. In this work, a solution is deemed infeasible if any of the tours is not completed within the specified time limit. The reliability is defined as the probability that a solution can be completed without failures. This work considers a STOP with dynamic rewards (STOPDR), in which the rewards associated with nodes have a static component and also a dynamic component. The dynamic component accounts for: (i) bonuses for visiting nodes early in a route; and (ii) penalties for nodes visited late in routes. The process through which bonus values and penalty values are generated is a hidden and unknown process. Hence, it has to be learnt from simulation observations. We consider the case in which bonuses are achieved for nodes visited as the first node in a route, while penalties are applied for nodes visited last in routes. The bonus and penalty values for nodes, when applicable, are assumed to follow an unknown statistical distribution. When solving the STOPDR, the challenge is to learn the bonuses and penalties associated with each node from the simulation testing of candidate solutions, while – at the same time – trying to maximize the total reward and also ensure that the solutions have a good level of reliability under stochastic travels times. Figure 1 provides an illustrative diagram depicting some of the main features of a STOPDR, including: stochastic edge traversal times;

A.1.3.2 Under Review

**SUPPORTING EFFICIENT ASSIGNMENT OF MEDICAL RESOURCES IN CANCER
TREATMENTS WITH SIMULATION-OPTIMIZATION**

Leandro do C. Martins
Juliana Castaneda
Angel A. Juan
Abtin Tondar
Laura Calvet

Barry B. Barrios

IN3 – Computer Science Dept.
Universitat Oberta de Catalunya
Rambla Poblenou 156
08018 Barcelona, SPAIN

Stanford University
Stanford Center for Professional Development
408 Panama Mall
Stanford, CA 94305, USA

Jose Luis Sanchez-Garcia

Hospital Universitario Vall d’Hebron
Universitat Autònoma de Barcelona
Passeig de la Vall d’Hebron, 119
08035 Barcelona, SPAIN

ABSTRACT

When scheduling multi-period medical treatments for patients with cancer, medical committees have to consider a large amount of data, variables, sanitary and budget constraints, as well as probabilistic elements. In many hospitals worldwide, medical specialists regularly decide the optimal schedule of treatments to be assigned to patients by considering multiple periods and the number of available resources. Hence, decisions have to be made upon the priority of each patient, available treatments, their expected effects, the proper order and intensity in which they should be applied. Consequently, medical experts have to assess many possible combinations and, eventually, choose the one that maximizes the survival chances or expected life quality of patients. To support this complex decision-making process, this paper introduces a novel methodology that combines a biased-randomized heuristic with simulation, to return ‘elite’ alternatives to experts. A simplified yet illustrative case study shows the main concepts and potential of the proposed approach.

1 INTRODUCTION

Cancers are medical conditions in which some cells divide uncontrollably and may invade their surrounding tissues. In many cancers, malignant cells can spread to other organs by entering the bloodstream and lymph systems. As well-documented in many official sources (e.g., www.cancer.gov), there are many types of cancers. For example, leukemia is a type of cancer in which the tissues in the body that make white blood cells (WBCs), such as the lymphatic system and bone marrow, create too many WBCs. Multiple myeloma and lymphoma are the cancers of plasma cells and lymphocytes, respectively. Squamous cell carcinoma

SOLVING A URBAN RIDESHARING PROBLEM WITH STOCHASTIC TRAVELING TIMES: A SIMHEURISTIC APPROACH

Leandro do C. Martins
Angel A. Juan

Universitat Oberta de Catalunya
IN3 – Computer Science Dept.
156 Rambla Poblenou
Barcelona, 08018, SPAIN

Maria Torres
Elena Perez-Bernabeu

Universitat Politècnica de València
Dept. of Applied Statistics and Operations Research
Plaza Ferrandiz y Carbonell
Alcoy, 03801, SPAIN

Canan G. Corlu

Metropolitan College
Boston University
1010 Commonwealth Avenue
Boston, MA 02215, USA

Javier Faulin

Institute of Smart Cities
Dept. Statistics, Computer Science, and Mathematics
Public University of Navarre
Pamplona, 31006, SPAIN

ABSTRACT

Ridesharing, carpooling, and carsharing concepts are redefining mobility practices in modern cities across the world. These concepts, however, also raise noticeable operational challenges that need to be efficiently addressed. In the urban ridesharing problem (URSP), a fleet of small private vehicles owned by citizens should be coordinated in order to pick up passengers on their way to work, hence maximizing the total value of their trips while not exceeding a deadline for reaching the destination points. Since this is a complex optimization problem, most of the existing literature assumes deterministic traveling times. This assumption is somewhat unrealistic and, for this reason, we discuss a richer URSP variant in which traveling times are modeled as random variables. Using random traveling times also forces us to consider a probabilistic constraint regarding the duration of each trip. For solving this stochastic optimization problem, an ‘agile’ simheuristic approach is proposed and tested via a series of computational experiments.

1 INTRODUCTION

During the last decades, a great increase in mobility demand has been requested worldwide, making people travels and goods movements much more frequent (Faulin et al. 2019). However, this demand cannot be always met, which has originated the concept of the sharing and gig economy. This concept is based on promoting collaboration among mobility agents which, in turn, allows for a reduction of mobility cost while still satisfying the initial demand level (Sawik et al. 2017). Another important benefit created by collaboration and ridesharing practices is the reduction in pollutant emissions. This is specially the case when internal combustion vehicles (ICV) are employed (Aloui et al. 2021; Brand et al. 2021; Pérez-Bernabeu et al. 2015), as it is still the case in many urban mobility operations. There are other mobility trends, such as ride-hailing – defined as the request for a car and driver to come immediately and take the customer to a previously fixed place –, that result on more comfortable trips, while also reducing parking

SUPPORTING HOSPITAL LOGISTICS DURING THE FIRST MONTHS OF THE COVID-19 CRISIS: A SIMHEURISTIC FOR THE STOCHASTIC TEAM ORIENTEERING PROBLEM

Markus Rabe
Jorge Chicaiza-Vaca

Rafael D. Tordecilla
Leandro do C. Martins
Angel A. Juan

Department of IT in Production and Logistics
Technical University Dortmund
Leonhard-Euler-Str. 5
Dortmund, 44227, GERMANY

IN3 – Computer Science Dept.
Universitat Oberta de Catalunya
Rambla Poblenou, 156
Barcelona, 08018, SPAIN

ABSTRACT

The unexpected crisis posed by the COVID-19 pandemic since March 2020 caused that items such as face shields, ear savers, and door openers were in high demand. In the area of Barcelona, thousands of volunteers employed their home 3D printers to produce these elements. Due to the lockdown, they had to be collected at each individual house by a reduced group of volunteer drivers, who transported them to several consolidation centers. These activities required a daily agile design of efficient routes, especially considering that drivers' exposure should be minimized – i.e., routes should not exceed a maximum time threshold. These constraints limit the number of individual houses that could be visited every day in order to collect newly produced items. Moreover, being a real-life environment, travel and service times are better modeled as random variables, which increases the problem complexity. This logistics challenge can be modeled as a stochastic team orienteering problem, with the objective of maximizing the total collected reward while satisfying the constraints on the fleet size and the maximum travel time per route. In order to solve this stochastic optimization problem, a simheuristic algorithm is proposed. Our approach, which also makes use of biased-randomization techniques, is able of generating high-quality solutions in short computing times.

1 INTRODUCTION

The COVID-19 pandemic crisis is one of the more recent greatest global challenges. The exponential increase in cases requiring medical care led to a sudden shortage of protective materials, putting medical and support staff at high risk of becoming infected as well. This not only jeopardized necessary attention in hospitals, but also accelerated the spread of COVID-19. Since March 2020, the pandemic has also had a strong impact in countries such as Germany and Spain. As in other regions, a community of volunteers called “Coronavirus Makers” was created in the Barcelona region to provide protective materials to staff in hospitals, nursing homes, and emergency medical care. The main tool was domestic 3D printers, which allowed a very quick design and elaboration of elements such as face shields, ear savers, or door openers. The bottleneck in this context was mainly a logistics one, as the lockdown meant that each 3D printer was in a single home, and collecting the items required optimizing the routing plans to maximize the added value of the items gathered while keeping drivers' safety. This paper describes the experience of bringing together different professional and personal profiles such as academics, volunteers, makers, and entrepreneurs, who typically employ different approaches when dealing with the pandemic. In this case, there was a need to find a quick way to apply knowledge accumulated over years of research to an urgent need, where every day counts. The goal was to support the Makers community in their volunteer initiative



ELSEVIER

Available online at www.sciencedirect.com

ScienceDirect

Transportation Research Procedia 00 (2021) 000–000

Transportation
Research
Procedia
www.elsevier.com/locate/procedia

XIV Transport Engineering Congress: 6th – 8th July 2021

An Agile and Reactive Biased-Randomized Heuristic for an Agri-Food Rich Vehicle Routing Problem

Rafael D. Tordecilla^{a,b,*}, Pedro J. Copado-Méndez^a, Javier Panadero^a, Leandro do C. Martins^a, Angel A. Juan^a

^aIN3 – Computer Science Dept., Universitat Oberta de Catalunya, Barcelona 08018, Spain

^bSchool of Engineering, Universidad de La Sabana, Chia 250001, Colombia

Abstract

Operational problems in agri-food supply chains usually show characteristics that are scarcely addressed by traditional academic approaches. These characteristics make an already NP-hard problem even more challenging; hence, this problem requires the use of tailor-made algorithms in order to solve it efficiently. This work addresses a rich vehicle routing problem in a real-world agri-food supply chain. Different types of animal food products are distributed to raising-pig farms. These products are incompatible, i.e., multi-compartment heterogeneous vehicles must be employed to perform the distribution activities. The problem considers constraints regarding visit priorities among farms, and not-allowed access of large vehicles to a subset of farms. Finally, a set of flat tariffs are employed to formulate the cost function. This problem is solved employing a reactive savings-based biased-randomized heuristic, which does not require any time-costly parameter fine-tuning process. Our results show savings in both cost and traveled distance when compared with the real supply chain performance.

© 2021 The Authors. Published by Elsevier B.V.

This is an open access article under the CC BY-NC-ND license (<http://creativecommons.org/licenses/by-nc-nd/4.0/>)

Peer-review under responsibility of the scientific committee of the 14th Conference on Transport Engineering.

Keywords: Rich Vehicle Routing Problem, Agri-Food Supply Chain, Biased-Randomized Heuristic.

1. Introduction

Feeding pigs in the pork production industry is a highly relevant activity to achieve successfully the supply chain goals (Rodríguez et al., 2014). Such activity requires a precise logistics from the production plant to the farms where the pigs are raised. Hence, our work consists in designing a set of vehicle routes that meet the feed demand of a set of pig farms, considering the real case of a pork production company in Spain. From an academic point of view, the analyzed problem can be considered as a rich vehicle routing problem (RVRP) (Caceres-Cruz et al., 2014), since: (i) vehicles are heterogeneous and have multiple compartments to separate different types of incompatible products that must be distributed to a set of farms; (ii) each farm may require multiple products; (iii) some farms admit only

* Corresponding author.

E-mail address: rtordecilla@uoc.edu, rafael.tordecilla@unisabana.edu.co



ELSEVIER

Available online at www.sciencedirect.com

ScienceDirect

Transportation Research Procedia 00 (2019) 000–000

Transportation
Research
Procedia
www.elsevier.com/locate/procedia

XIV Transport Engineering Congress: 6th – 8th July 2021

Using Data Analytics & Machine Learning to Design Business Interruption Insurance Products for Rail Freight Operators

John F. Cardona^{a,*}, Juliana Castaneda^a, Leandro do C. Martins^a, Mariem Gandouz^a,
Guillermo Franco^b, Angel A. Juan^a

^aIN3 – Computer Science Dept., Universitat Oberta de Catalunya, Av. Carl Friedrich Gauss 5, 08860 Castelldefels, Spain

^bGuy Carpenter Inc., New York, NY, USA

Abstract

This paper discusses a case study in which publicly available data on a rail freight transportation firm has been gathered, cleansed, and analyzed in order to: (i) describe the data using statistical indicators and graphs; (ii) identify patterns regarding several Key Performance Indicators; (iii) obtain forecasts on the future evolution of these indicators; and (iv) use the identified patterns and the generated forecasts to propose customized insurance products that reflect the current and future freight transportation activity. The paper illustrates the different methodological steps required during the extraction and cleansing of the data –which required the development of Python scripts–, the use of time series analysis for obtaining reliable forecasts, and the use of machine learning models for designing customized insurance coverage from the identified patterns and predicted values.

© 2020 The Authors. Published by Elsevier B.V.

This is an open access article under the CC BY-NC-ND license (<http://creativecommons.org/licenses/by-nc-nd/4.0/>)

Peer-review under responsibility of the scientific committee of the 14th Conference on Transport Engineering..

Keywords: Data analytics; machine learning; business interruption; insurance, rail freight

1. Introduction

Data has become an essential element for the operational development and economic growth of many organizations throughout the world. As organizations across the world reach a level of economic abundance, their capacity to efficiently and effectively manage data has become a matter for concern (Parr-Rud, 2012). As a result, organizations from all industries and fields, and those which contribute to an emerging economy are welcoming innovative solutions and methods for managing their volumes of data. For instance, organizations from the railroad freight transportation sector in the United States (U.S.) have experienced substantial data growth throughout the years. This valuable information can be used for identifying factors which can prevent an organization from reaching and maintaining economic growth. According to Samson and Previts (1999), the first train freight transport service contracted in the U.S. took place in 1827. Shipments of cargo were less frequent, and hence, the volume of standard goods being

* Corresponding author.

E-mail address: jcardonadi@uoc.edu



ELSEVIER

Available online at www.sciencedirect.com

ScienceDirect

Transportation Research Procedia 00 (2019) 000–000

Transportation
Research
Procedia
www.elsevier.com/locate/procedia

XIV Transport Engineering Congress: 6th – 8th July 2021

Supervised Machine Learning Algorithms for Measuring and Promoting Sustainable Transportation and Green Logistics

Juliana Castaneda*, John F. Cardona, Leandro do C. Martins, Angel A. Juan

IN3 – Computer Science Dept., Universitat Oberta de Catalunya, Av. Carl Friedrich Gauss 5, 08860 Castelldefels, Spain

Abstract

The sustainable development of freight transport has received much attention in recent years. The new regulations for sustainable transport activities established by the European Commission and the United Nations have created the need for road freight transport companies to develop methodologies to measure the social and environmental impact of their activities. This work aims to develop a model based on supervised machine learning methods with intelligent classification algorithms and key performance indicators for each dimension of sustainability as input data. This model allows establishing the level of sustainability (high, medium or low). Several classification algorithms were trained, finding that the support vector machines algorithm is the most accurate, with 98% accuracy for the data set used. The model is tested by establishing the level of sustainability of a European company in the road freight sector, thus allowing the establishment of green strategies for its sustainable development.

© 2021 The Authors. Published by Elsevier B.V.

This is an open access article under the CC BY-NC-ND license (<http://creativecommons.org/licenses/by-nc-nd/4.0/>)

Peer-review under responsibility of the scientific committee of the 14th Conference on Transport Engineering.

Keywords: Machine Learning; Freight Transportation; Sustainability; Classification Algorithms.

1. Introduction

The growing concern about climate change has impacted people and businesses, making sustainability a trend in all economic activities around the world. The constant design of strategies to mitigate the environmental damage generated by humans' activities on the planet is more than a trend. In the business world, it is becoming a requirement. Integrating technologies to measure the impact of their activities leads to control over them and supports the strategies established to alleviate the generated impact. Freight transport in the European Union has been growing significantly in the last decade. In 2017, it registered a total increase of 2.4 %, compared to 2016, being road freight transport (RFT) the main contributor with +4.7% (EEA, 2019). This means an increase of the demand for services in freight transport caused by the development of the global trade and influenced by the consumerism of the society (Nowakowska-Grunt and Strzelczyk, 2019; Nowicka-Skowron and Mesjasz-Lech, 2013). RFT is the main source of greenhouse gas (GHG)

* Corresponding author.

E-mail address: jcastanedaji@uoc.edu

B.2 Complementary Information

B.2.1 Mixed Integer Linear Programming Model of the Vehicle Routing Problem with Optional Backhauls

This Section introduces a MILP model of the VRPOB, introduced in Section 3.2. Firstly the sets and parameters are defined, thence the decision variables, objective function, and constraints.

Sets	
V	: The set of all nodes including the depot, linehaul and backhaul customers.
L	: The set of linehaul customers.
B	: The set of backhaul customers.
A	: The set of edges linking each pair of nodes.
K	: The set of homogeneous vehicles.
Parameters	
d_i	: Demand of customer $i \in V$.
c_{ij}	: Cost of travel from node $i \in V$ to node $j \in V$.
h_i	: Unitary penalty cost (e.g., unitary inventory cost) per RTI not collected from customer $i \in B$.
q	: Capacity of each vehicle.
Variables	
x_{ijk}	: Binary variable that takes the value 1 if edge $(i, j) : i \in L \cup \{0\}, j \in L$ is on the linehaul route traveled by vehicle $k \in K$, being 0 otherwise.
y_{ijk}	: Binary variable that takes the value 1 if edge $(i, j) : i \in L \cup B, j \in B \cup \{0\}$ is on the backhaul route traveled by vehicle $k \in K$, being 0 otherwise.
z_{ik}	: Binary variable that takes the value 1 if customer $i \in V \setminus \{0\}$ is visited by vehicle $k \in K$, being 0 otherwise.
f_{ijk}	: Variable to eliminate subtours in the linehaul route visited by vehicle $k \in K$ and for each edge $(i, j) : i \in L \cup \{0\}, j \in L$.
g_{ijk}	: Variable to eliminate subtours in the backhaul route visited by vehicle $k \in K$ and for each edge $(i, j) : i \in L \cup B, j \in B \cup \{0\}$.

B.2.1.1 Mathematical Model

Minimize:

$$\sum_{i \in B} h_i d_i \left(1 - \sum_{k \in K} z_{ik} \right) + \sum_{k \in K} \sum_{i \in L \cup \{0\}} \sum_{j \in L, i \neq j} c_{ij} x_{ijk} + \sum_{k \in K} \sum_{i \in L \cup B} \sum_{j \in B \cup \{0\}, i \neq j} c_{ij} y_{ijk} \quad (1)$$

subject to:

$$\sum_{k \in K} z_{ik} = 1, \quad \forall i \in L \quad (2)$$

$$\sum_{k \in K} z_{ik} \leq 1, \forall i \in B \quad (3)$$

$$\sum_{j \in L} x_{0jk} = 1, \forall k \in K \quad (4)$$

$$\sum_{i \in V} y_{i0k} = 1, \forall k \in K \quad (5)$$

$$\sum_{i \in L \cup \{0\}, i \neq h} x_{ihk} + \sum_{j \in L, j \neq h} x_{hjk} + \sum_{j \in B \cup \{0\}, j \neq h} y_{hjk} = 2z_{hk}, \forall h \in L, \forall k \in K \quad (6)$$

$$\sum_{i \in L \cup B, i \neq h} y_{ihk} + \sum_{j \in B \cup \{0\}, j \neq h} y_{hjk} = 2z_{hk}, \forall h \in B, \forall k \in K \quad (7)$$

$$\sum_{i \in L \cup \{0\}, i \neq h} x_{ihk} = \sum_{j \in L, j \neq h} x_{hjk} + \sum_{j \in B \cup \{0\}, j \neq h} y_{hjk} \quad \forall h \in L, \forall k \in K \quad (8)$$

$$\sum_{i \in L \cup B, i \neq h} y_{ihk} = \sum_{j \in B \cup \{0\}, j \neq h} y_{hjk}, \quad \forall h \in B, \forall k \in K \quad (9)$$

$$\sum_{j \in L, j \neq h} x_{hjk} + \sum_{j \in B \cup \{0\}, j \neq h} y_{hjk} \leq 1, \quad \forall h \in L, \forall k \in K \quad (10)$$

$$\sum_{i \in B \cup \{0\}, i \neq h} y_{ihk} \leq 1, \quad \forall h \in B, \forall k \in K \quad (11)$$

$$\sum_{i \in L} d_i z_{ik} \leq q, \quad \forall k \in K \quad (12)$$

$$\sum_{i \in B} d_i z_{ik} \leq q, \quad \forall k \in K \quad (13)$$

$$f_{ijk} \leq n x_{ijk}, \quad \forall i \in L \cup \{0\}, \forall j \in L, i \neq j, \forall k \in K \quad (14)$$

$$\sum_{i \in L \cup \{0\}, i \neq j} f_{ijk} - \sum_{h \in L, h \neq j} f_{jhk} = z_{jk}, \quad \forall j \in L, \forall k \in K \quad (15)$$

$$\sum_{j \in L} f_{0jk} = \sum_{j \in L} z_{jk}, \quad \forall k \in K \quad (16)$$

$$g_{ijk} \leq m y_{ijk}, \quad \forall i \in L \cup B, \forall j \in B \cup \{0\}, i \neq j, \forall k \in K \quad (17)$$

$$\sum_{h \in B \cup \{0\}, h \neq j} g_{jhk} - \sum_{i \in L \cup B, i \neq j} g_{ijk} = z_{jk}, \quad \forall j \in B, \forall k \in K \quad (18)$$

$$\sum_{j \in B} g_{j0k} = \sum_{j \in B} z_{jk}, \quad \forall k \in K \quad (19)$$

$$x_{ijk}, y_{ijk}, z_{ik} \in \{0, 1\}, \forall i, j \in V, k \in K \quad (20)$$

$$f_{ijk}, g_{ijk} \in \mathbb{Z}^+, \forall i, j \in V, k \in K \quad (21)$$

In this model, Equation (1) minimizes the total routing plus penalty cost. Constraints (2) ensure that every LH customer is visited by only one vehicle. Constraints (3) make certain that at most only one vehicle visits a BH customer. Constraints (4) and (5) warrant that each vehicle leaves and returns to the depot. Constraints (6) and (7) certify that two edges (one entering and one leaving) are assigned to a customer only if this is serviced. Constraints (8) and (9) establish that if one vehicle enters a customer node, it must departure from it as well. Constraints (10) assure that after departing from each LH customer, a vehicle must either serve another LH customer or make an empty trip either to the depot or to a BH customer. Constraints (11) make sure that before a vehicle visits a BH customer, it must have serviced either another BH customer or a LH customer. Constraints (12) and (13) ensure that the total quantity of product to deliver or pickup does not exceed a single vehicle capacity, both in LH and BH groups, respectively. Based on the formulation proposed by Gavish and Graves (1978), constraints (14), (15), and (16) eliminate subtours for LH customers. Likewise, constraints (17), (18), and (19) do the same regarding subtours for BH customers. Other formulations for subtour elimination can be found in Öncan et al. (2009). Finally, constraints (20) and (21) indicate the variables that are binary and integer, respectively.

B.2.2 Mixed Integer Linear Programming Model of the Hybrid Flow-Shop Vehicle Routing Problem

This Section introduces a MILP model of the HFS-VRP, stated in Section 6.1.

Sets	
J	: The set of jobs $\{1 \dots n\}$.
S	: The set of stages $\{1 \dots s\}$.
M_s	: The set of machines at stage $s \in S$ $\{1 \dots m_s\}$.
R	: The set of trips (deliveries of batches) $\{1 \dots r\}$.
C	: The set of customers $\{1 \dots c\}$.
JC_c	: The set of jobs of customer $c \in C$ $\{1 \dots jc_c\}$.
Parameters	
B	: A very big constant.
$P_{j,s}$: The processing time of job $j \in J$ at stage $s \in S$.
Q	: The capacity of <i>vehicle</i> .
q_j	: The loading capacity required by job $j \in J$.
$TT_{c,a}$: The travel time between customer $c \in C \cup \{0\}$ and $a \in C \cup \{0\}$ (where node 0 is the factory).
Variables	

$X_{j,h,s}$: Binary variable that takes the value of 1 if job $j \in J$ is processed before job $h \in J$ at stage $s \in S$, and 0, otherwise.
$Y_{j,s,m}$: Binary variable that takes the value of 1 if job $j \in J$ is processed on machine $m \in E_s$ of stage $s \in S$, and 0 otherwise.
$ST_{j,s}$: Continuous variable for the starting time of job $j \in J$ processed on machine $m \in E_s$ of stage $s \in S$.
$CT_{j,s}$: Continuous variable for the completion time of job $j \in J$ processed on machine $m \in E_s$ of stage $s \in S$.
SR_r	: The departure time of trip $r \in R$ of the vehicle.
CR_r	: The completion time of trip $r \in R$ of the vehicle.
$TV_{c,r}$: The time of arrival at customer $c \in C \cup \{0\}$ on trip $r \in R$.
$F_{c,a,r}$: Binary variable that takes the value of 1 if customer $c \in C \cup \{0\}$ is visited before customer $a \in C \cup \{0\}$ in trip $r \in R$.
$W_{j,r}$: Binary variable that takes the value of 1 if job $j \in J$ is dispatched on trip $r \in R$.
G_r	: Binary variable that takes the value of 1 if the vehicle performs the trip $r \in R$.
$N_{c,r}$: Binary variable that takes the value of 1 if the customer $c \in C$ is visited on trip $r \in R$.
$Cmax$: The makespan or maximum dispatching time of the jobs.

B.2.2.1 Mathematical Model

Minimize:

$$C_{max} \quad (22)$$

subject to:

$$\sum_{m \in M_s} Y_{j,s,m} = 1 \quad \forall j \in J, \forall s \in S \quad (23)$$

$$CT_{j,s} = ST_{j,s} + P_{j,s} \quad \forall j \in J, \forall s \in S, \forall m \in M_s \quad (24)$$

$$ST_{j,s} \geq CT_{j,s-1} \quad \forall j \in J, \forall s \in S, s > 1 \quad (25)$$

$$ST_{h,s} \geq CT_{j,s} - B \cdot (3 - X_{j,h,s} - Y_{j,s,m} - Y_{h,s,m}) \quad \forall j, h \in J, \forall s \in S, \forall m \in M_s, j \neq h \quad (26)$$

$$ST_{j,s} \geq CT_{h,s} - B \cdot X_{j,h,s} - B \cdot (2 - Y_{j,s,m} - Y_{h,s,m}) \quad \forall j, h \in J, \forall s \in S, \forall m \in M_s, j \neq h \quad (27)$$

$$SR_r \geq CT_{j|S|} - B \cdot (1 - W_{j,r}) \quad \forall j \in J, \forall r \in R \quad (28)$$

$$TV_{c,r} \geq TV_{a,r} + TT_{a,c} - B \cdot (1 - F_{j,r}) \quad \forall c \in C, \forall a \in C \cup \{0\}, \forall r \in R, c \neq a \quad (29)$$

$$TV_{0,r} \geq SR_r \quad \forall r \in R \quad (30)$$

$$\sum_{j \in J_c} W_{j,r} \leq N_{c,r} \cdot B \quad \forall c \in C, \forall r \in R \quad (31)$$

$$N_{c,r} \leq \sum_{j \in J_c} W_{j,r} \quad \forall c \in C, \forall r \in R \quad (32)$$

$$\sum_{a \in C \cup \{0\}, a \neq c} F_{a,c,r} = N_{c,r} \quad \forall c \in C, \forall r \in R \quad (33)$$

$$\sum_{a \in C \cup \{0\}, a \neq c} F_{c,a,r} = N_{c,r} \quad \forall c \in C, \forall r \in R \quad (34)$$

$$\sum_{c \in C} F_{0,c,r} = G_{c,r} \quad \forall r \in R \quad (35)$$

$$\sum_{c \in C} F_{c,0,r} = G_{c,r} \quad \forall r \in R \quad (36)$$

$$\sum_{r \in R} W_{j,r} = 1 \quad \forall j \in J \quad (37)$$

$$\sum_{j \in J} q_j \cdot W_{j,r} \leq Q \cdot G_r \quad \forall r \in R \quad (38)$$

$$CR_r \geq TV_r \quad \forall c \in C, \forall r \in R \quad (39)$$

$$SR_{r+1} \geq CR_r + TT_{c,0} - B \cdot (1 - F_{c,0,r}) \quad \forall c \in C, \forall r \in R, r < |R| \quad (40)$$

$$C_{max} \geq CR_r \quad \forall r \in R \quad (41)$$

$$G_r \leq G_{r-1} \quad \forall r \in R, r > 1 \quad (42)$$

$$X_{j,h,s} \in \{0,1\} \quad \forall j, h \in J, \forall s \in S \quad (43)$$

$$Y_{j,s,m} \in \{0,1\} \quad \forall j \in J, \forall s \in S, \forall m \in M_s \quad (44)$$

$$F_{c,a,r} \in \{0,1\} \quad \forall c \in C \cup \{0\}, \forall a \in C \cup \{0\}, \forall r \in R \quad (45)$$

$$W_{j,r} \in \{0,1\} \quad \forall j \in J, \forall r \in R \quad (46)$$

$$G_r \in \{0,1\} \quad \forall r \in R \quad (47)$$

$$N_{c,r} \in \{0,1\} \quad \forall c \in C, \forall r \in R \quad (48)$$

$$CT_{j,s} \geq 0 \quad \forall j \in J, \forall s \in S \quad (49)$$

$$ST_{j,s} \geq 0 \quad \forall j \in J, \forall s \in S \quad (50)$$

$$SR_r \geq 0 \quad \forall r \in R \quad (51)$$

$$CR_r \geq 0 \quad \forall r \in R \quad (52)$$

$$TV_{c,r} \geq 0 \quad \forall c \in C \cup \{0\}, \forall r \in R \quad (53)$$

Equation (22) represents the objective function, that is the minimization of the makespan (the time in which the last job is delivered). Constraints set (23) specifies that each job can be assigned at only one machine at each stage. Constraints set (24) calculates the completion time of each job at each stage. Constraints set (25) determines the minimum starting time of each job at each stage regarding the completion time of the job in the previous stage. Constraints sets (26) and (27) specify the minimum starting time of each job at each stage

regarding the completion time of jobs processed before at the same machine. Constraints set (28) defines the minimum starting time of each trip regarding the maximum completion time of the jobs that are going to be dispatched on that trip. Constraints set (29) specifies the minimum time of the visit of a customer in a trip depending on the time of the visit of the previous customer in that trip, and the travel time between both customers. Constraints set (30) indicates that the time of the visit of the depot (node 0) in a trip is equal to the departure time of that trip. Constraints sets (31) and (32) guarantee that, if a job is dispatched on a trip, the customer who is the owner of that job is visited on that trip. Constraints sets (33) and (34) state that, if a customer is visited on a trip, that customer is a successor and a predecessor of another customer or depot. Constraints sets (35) and (36) ensure that each trip starts and ends at depot if the trip is performed (node 0). Constraints set (37) guarantees that each job is dispatched in exactly one trip. Constraints set (38) assure that the volume capacity of the vehicle on each trip is not surpassed. Constraints set (39) calculates the completion time of a trip regarding the time of the last customer visited in that trip. Constraints set (40) states that the starting time of a trip is greater or equal than the return time of the vehicle to the depot after the previous trip. Constraints set (41) specifies that the completion time of the last delivery is greater or equal than the completion time of the last trip. Constraints set (42) controls the binary variables of trips, ensuring that only consecutive trips can be performed. Finally, constraints sets (43)-(53) define the domain of decision variables.

B.2.2.2 A First Lower Bound for the HFS-VRP

Considering the NP-hardness of the problem, a first lower bound calculation is proposed for the problem, $LB_{HFS-VRP}$, in order to evaluate the performance of our proposed algorithms. Since the deliveries of some finished jobs can be performed simultaneously with the production of other jobs, it can be said that the HFS stage is overlapped partially with the VRP stage. For that reason, the proposed $LB_{HFS-VRP}$ consists in the maximum of the following two partial lower bounds (54):

- i. The $LB_{HFS-VRP_{part1}}$ (Equation 55), which consists in adding the HFS lower bound proposed by Haouari and Hidri, 2008 (56) and the traveling time of the nearest customer to the factory.
- ii. The $LB_{HFS-VRP_{part2}}$ (Equation 63), which is obtained by the aggregation of a proposed lower bound for the multi-trip single VRP and the minimum summation of processing times across all jobs.

$$LB_{HFS-VRP} = \max\{LB_{HFS-VRP_{part1}}, LB_{HFS-VRP_{part2}}\} \quad (54)$$

As stated, Haouari and Hidri (2008) proposed a HFS lower bound. This bound summed with the smallest traveling time from the factory to a customer gives a possible lower bound for the HFS-VRP (55). Equations 56-62, which were taken from Haouari and Hidri (2008), supports the calculation of $LB_{HFS-VRP_{part1}}$.

$$LB_{HFS-VRP_{part1}} = LB_{HFS} + \min_{c \in C} \{TT_{0,c}\} \quad (55)$$

$$LB_{HFS} = \max_{2 \leq s \leq |S|} \{LB'_s\} \quad (56)$$

$$LB'_s = JL_{1,s-1} + \frac{SPT_{s-1}(|M_s|) + \sum_{j \in J} P_{j,s} + \sum_{k \in M_s} JR_{k,s}}{|M_s|} \quad (57)$$

$$LS_{j,s} = \begin{cases} \sum_{k=1}^{s-1} P_{j,k} & \text{if } j \in J, s > 1 \\ 0 & \text{if } j \in J, s = 1 \end{cases} \quad (58)$$

$$RS_{j,s} = \begin{cases} \sum_{k=s+1}^{|S|} P_{j,k} & \text{if } j \in J, j < s \\ 0 & \text{if } j \in J, s = |S| \end{cases} \quad (59)$$

$$JL_{l,s}: \text{ the } l\text{th smallest value of } LS_{j,s} \quad (60)$$

$$JR_{l,s}: \text{ the } l\text{th smallest value of } RS_{j,s} \quad (61)$$

$$SPT_{l,s}(k): \text{ the minimum-sum of completion times of the } k \text{ smallest } (s-1)\text{-stage jobs } LS_{j,s} \quad (62)$$

The lower bound that is proposed for the multi-trip single VRP $LB_{HFS-VRP_{part2}}$ (63) is constructed considering that:

- i. The total departing times from the depot to the first customer of the trips should be at least the minimum distance from the factory to a customer multiplied by the number of trips.
- ii. The total traveling time of the vehicle should be at least the minimum travel time from the factory to a customer multiplied by two times the number of trips (departure and return of each trip). Nevertheless, this multiplication considers the last return to the factory, thence this distance should be subtracted once. Therefore, the total traveling time of the vehicle should be at least two times the minimum travel time from the factory to a customer times the minimum number of trips minus the minimum travel time from the factory to a customer.
- iii. The number of arcs visited between customers (that does not include the arcs that connect with the depot) is at least the number of customers minus the minimum number of trips $|C| - MNT$. Thence, the total traveling time across arcs is at least the sum of $|C| - MNT$ smallest distances between customers.
- iv. If the vehicle only has to do only one trip, then the traveling time is at least the sum of the minimum travel time from the factory to a customer with the $|C| - 1$ smallest distances between customers and with the LB_{HFS} .

Equations (64)-(67) support the calculation of $LB_{HFS-VRP_{part2}}$.

$$LB_{HFS-VRP_{part2}} = \begin{cases} \min_{c \in C} TT_{0,c} + \sum_{i=1}^{|C|-1} STTO_i + LB_{HFS} & \text{if } MNT = 1 \\ 2 \cdot MNT \cdot \min_{c \in C} TT_{0,c} - \min_{c \in C} TT_{0,c} + \min_{j \in J} SUMPT_j & \text{if } MNT \geq |C|, MNT > 1 \\ 2 \cdot MNT \cdot \min_{c \in C} TT_{0,c} - \min_{c \in C} TT_{0,c} + \sum_{i=1}^{|C|-MNT} STTO_i + \min_{j \in J} SUMPT_j & \text{if } |C| > MNT > 1 \end{cases} \quad (63)$$

$$MNT = \frac{\sum_j in_j q_j}{Q}: \text{the minimum trips of the vehicle} \quad (64)$$

$$STT_c = \min_{h \in C, h \neq c} \{TT_{c,h}\} \quad (65)$$

$$STTO_l: \text{the } l\text{-th smallest traveling time of } STT \text{ values} \quad (66)$$

$$SUMPT_j = \sum_{s \in S} P_{j,s} \quad (67)$$

Bibliography

- Abdulkader, Mohamed Mahmoud Saleh, Yuvraj Gajpal, and Tarek Y ElMekkawy (2018). "Vehicle routing problem in omni-channel retailing distribution systems". In: *International Journal of Production Economics* 196, pp. 43–55.
- Adomavicius, Gediminas and Alexander Tuzhilin (2005). "Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions". In: *IEEE Transactions on Knowledge & Data Engineering* 17.6, pp. 734–749.
- Afshar, Abbas and Ali Haghani (2012). "Modeling integrated supply chain logistics in real-time large-scale disaster relief operations". In: *Socio-Economic Planning Sciences* 46.4, pp. 327–338.
- Agatz, Niels, Alan Erera, Martin Savelsbergh, and Xing Wang (2012). "Optimization for dynamic ride-sharing: A review". In: *European Journal of Operational Research* 223.2, pp. 295–303.
- Agatz, Niels A.H., Alan L. Erera, Martin W.P. Savelsbergh, and Xing Wang (2011). "Dynamic ride-sharing: A simulation study in metro Atlanta". In: *Transportation Research Part B: Methodological* 45.9, pp. 1450–1464.
- Agnihotri, Arpita (2015). "Can brick-and-mortar retailers successfully become multichannel retailers?" In: *Journal of Marketing Channels* 22.1, pp. 62–73.
- Ahn, Hyung Jun (2008). "A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem". In: *Information Sciences* 178.1, pp. 37–51.
- Ahn, Sang, Colin Cooper, Gerard Cornuejols, and Alan Frieze (1988). "Probabilistic analysis of a relaxation for the k-median problem". In: *Mathematics of Operations Research* 13.1, pp. 1–31.
- Ai, The Jin and Voratas Kachitvichyanukul (2009). "A particle swarm optimization for the vehicle routing problem with simultaneous pickup and delivery". In: *Computers & Operations Research* 36.5, pp. 1693–1702.
- Aissat, Kamel and Ammar Oulamara (2014). "A priori approach of real-time ridesharing problem with intermediate meeting locations". In: *Journal of Artificial Intelligence and Soft Computing Research* 4.4, pp. 287–299.
- Al Chami, Zaher, Hamza El Flity, Hervé Manier, and Marie-Ange Manier (2018). "A new metaheuristic to solve a selective pickup and delivery problem". In: *2018 4th International Conference on Logistics Operations Management (GOL)*. IEEE, pp. 1–5.
- Albino, Vito, Umberto Berardi, and Rosa Maria Dangelico (2015). "Smart cities: Definitions, dimensions, performance, and initiatives". In: *Journal of Urban Technology* 22.1, pp. 3–21.
- Alonso-Mora, Javier, Samitha Samaranayake, Alex Wallar, Emilio Frazzoli, and Daniela Rus (2017). "On-demand high-capacity ride-sharing via dynamic trip-vehicle assignment". In: *Proceedings of the National Academy of Sciences* 114.3, pp. 462–467.
- Alvarez Fernandez, Stephanie, Daniele Ferone, Angel Juan, and Daniele Tarchi (2021). "A simheuristic algorithm for video streaming flows optimisation with QoS threshold modelled as a stochastic single-allocation p-hub median problem". In: *Journal of Simulation*, pp. 1–14.
- Andrade, Carlos E, Thuener Silva, and Luciana S Pessoa (2019). "Minimizing flowtime in a flow-shop scheduling problem with a biased random-key genetic algorithm". In: *Expert Systems with Applications* 128, pp. 67–80.

- Ansari, Abdollah, Ibrahim Said Ahmad, Azuraliza Abu Bakar, and Mohd Ridzwan Yaakub (2020). "A Hybrid Metaheuristic Method in Training Artificial Neural Network for Bankruptcy Prediction". In: *IEEE Access* 8, pp. 176640–176650.
- Arab, R, SF Ghaderi, and R Tavakkoli-Moghaddam (2018). "Bi-objective inventory routing problem with backhauls under transportation risks: two meta-heuristics". In: *Transportation Letters*, pp. 1–17.
- Armstrong, Ronald, Su Gao, and Lei Lei (2008). "A zero-inventory production and distribution problem with a fixed customer sequence". In: *Annals of Operations Research* 159.1, pp. 395–414.
- Arnau, Quim, Angel A Juan, and Isabel Serra (2018). "On the use of learnheuristics in vehicle routing optimization problems with dynamic inputs". In: *Algorithms* 11.12, p. 208.
- Assis, Luciana P, André L Maravilha, Alessandro Vivas, Felipe Campelo, and Jaime A Ramírez (2013). "Multiobjective vehicle routing problem with fixed delivery and optional collections". In: *Optimization letters* 7.7, pp. 1419–1431.
- Avci, Mustafa and Seyda Topaloglu (2016). "A hybrid metaheuristic algorithm for heterogeneous vehicle routing problem with simultaneous pickup and delivery". In: *Expert Systems with Applications* 53, pp. 160–171.
- Aziez, Imadeddine, Jean-François Côté, and Leandro C Coelho (2020). "Exact algorithms for the multi-pickup and delivery problem with time windows". In: *European Journal of Operational Research*.
- Balakrishnan, Jaydeep, Chun Hung Cheng, Kam Fai Wong, and Kwan Ho Woo (2018). "Product recommendation algorithms in the age of omnichannel retailing – An intuitive clustering approach". In: *Computers & Industrial Engineering* 115. December 2017, pp. 459–470.
- Balasingam, Manohari (2017). "Drones in medicine – The rise of the machines". In: *International Journal of Clinical Practice* 71.9, e12989.
- Baldacci, Roberto, Vittorio Maniezzo, and Aristide Mingozzi (2004). "An exact method for the car pooling problem based on lagrangean column generation". In: *Operations Research* 52.3, pp. 422–439.
- Balinski, Michel Louis (1966). "On finding integer solutions to linear programs". In: *Proceedings of IBM scientific symposium on combinatorial problems*. IBM White Plains, NY, pp. 225–248.
- Ballús-Armet, Ingrid, Susan A Shaheen, Kelly Clonts, and David Weinzimmer (2014). "Peer-to-peer carsharing: Exploring public perception and market characteristics in the San Francisco Bay area, California". In: *Transportation Research Record* 2416.1, pp. 27–36.
- Bamburly, Dane (2015). "Drones: Designed for product delivery". In: *Design Management Review* 26.1, pp. 40–48.
- Barahona, Francisco and Fabián A Chudak (2005). "Near-optimal solutions to large-scale facility location problems". In: *Discrete Optimization* 2.1, pp. 35–50.
- Barbarosoglu, Gulay and Demet Ozgur (1999). "A tabu search algorithm for the vehicle routing problem". In: *Computers & Operations Research* 26.3, pp. 255–270.
- Baykal, M. O., R. Alhadj, and F. Polat (2005). "Co-operation framework of case-based reasoning agents for automated product recommendation". In: *Journal of Experimental and Theoretical Artificial Intelligence* 17.3, pp. 201–220.
- Bayliss, Christopher, Angel A Juan, Christine SM Currie, and Javier Panadero (2020a). "A learnheuristic approach for the team orienteering problem with aerial drone motion constraints". In: *Applied Soft Computing*, p. 106280.
- Bayliss, Christopher, Leandro do C Martins, and Angel A Juan (2020b). "A Two-phase Local Search with a Discrete-event Heuristic for the Omnichannel Vehicle Routing Problem". In: *Computers & Industrial Engineering*, p. 106695.

- Beck, Norbert and David Rygl (2015). "Categorization of multiple channel retailing in Multi-, Cross-, and Omni-Channel Retailing for retailers and retailing". In: *Journal of Retailing and Consumer Services* 27, pp. 170–178.
- Belgin, Onder, Ismail Karaoglan, and Fulya Altiparmak (2018). "Two-echelon vehicle routing problem with simultaneous pickup and delivery: Mathematical model and heuristic approach". In: *Computers & Industrial Engineering* 115, pp. 1–16.
- Belloso, Javier, Angel A Juan, and Javier Faulin (2019). "An iterative biased-randomized heuristic for the fleet size and mix vehicle-routing problem with backhauls". In: *International Transactions in Operational Research* 26.1, pp. 289–301.
- Belloso, Javier, Angel A Juan, Enoc Martinez, and Javier Faulin (2017). "A biased-randomized meta-heuristic for the vehicle routing problem with clustered and mixed backhauls". In: *Networks* 69.3, pp. 241–255.
- Berbeglia, Gerardo, Jean-François Cordeau, Irina Gribkovskaia, and Gilbert Laporte (2007). "Static pickup and delivery problems: a classification scheme and survey". In: *Top* 15.1, pp. 1–31.
- Bernstein, Fernando and Victor Martínez-de Albéniz (2017). "Dynamic product rotation in the presence of strategic customers". In: *Management Science* 63.7, pp. 2092–2107.
- Bettinelli, Andrea, Alberto Ceselli, and Giovanni Righini (2014). "A branch-and-price algorithm for the multi-depot heterogeneous-fleet pickup and delivery problem with soft time windows". In: *Mathematical Programming Computation* 6.2, pp. 171–197.
- Bianchessi, Nicola and Giovanni Righini (2007). "Heuristic algorithms for the vehicle routing problem with simultaneous pick-up and delivery". In: *Computers & Operations Research* 34.2, pp. 578–594.
- Bianchi-Aguiar, Teresa, Elsa Silva, Luis Guimarães, Maria Antónia Carravilla, José F Oliveira, João Günther Amaral, Jorge Liz, and Sérgio Lapela (2016). "Using Analytics to Enhance a Food Retailer's Shelf-Space Management". In: *Interfaces* 46.5, pp. 424–444.
- Bibri, Simon Elias and John Krogstie (2019). "Generating a vision for smart sustainable cities of the future: a scholarly backcasting approach". In: *European Journal of Futures Research* 7.1, p. 5.
- Bilde, Ole and Jakob Krarup (1977). "Sharp Lower Bounds and Efficient Algorithms for the Simple Plant Location Problem". In: *Studies in Integer Programming*. Ed. by P.L. Hammer, E.L. Johnson, B.H. Korte, and G.L. Nemhauser. Vol. 1. Annals of Discrete Mathematics. Amsterdam, Netherlands: Elsevier, pp. 79–97.
- Bistaffa, Filippo, Christian Blum, Jesús Cerquides, Alessandro Farinelli, and Juan A Rodríguez-Aguilar (2019). "A Computational Approach to Quantify the Benefits of Ridesharing for Policy Makers and Travellers". In: *IEEE Transactions on Intelligent Transportation Systems*, pp. 1–12.
- Blau, Peter (2017). *Exchange and power in social life*. Routledge.
- Blum, Christian and Andrea Roli (2003). "Metaheuristics in combinatorial optimization: Overview and conceptual comparison". In: *ACM computing surveys (CSUR)* 35.3, pp. 268–308.
- Bonami, Pierre and Jon Lee (2007). "BONMIN user's manual". In: *Numer Math* 4, pp. 1–32.
- Bongiovanni, Claudia, Mor Kaspi, and Nikolas Geroliminis (2019). "The electric autonomous dial-a-ride problem". In: *Transportation Research Part B: Methodological* 122, pp. 436–456.
- Boonmee, Chawis, Mikiharu Arimura, and Takumi Asada (2017). "Facility location optimization model for emergency humanitarian logistics". In: *International Journal of Disaster Risk Reduction* 24, pp. 485–498.
- Borcuch, Artur (2016). "The sharing economy: Understanding and challenges". In: *International Journal of Humanities & Social Science Studies (IJHSSS)*.
- Bozorgchenani, Arash, Daniele Tarchi, and Giovanni Emanuele Corazza (2018a). "A control and data plane split approach for partial offloading in mobile fog networks". In: *2018 IEEE Wireless Communications and Networking Conference (WCNC)*. IEEE, pp. 1–6.

- Bozorgchenani, Arash, Daniele Tarchi, and Giovanni Emanuele Corazza (2018b). "Mobile edge computing partial offloading techniques for mobile urban scenarios". In: *2018 IEEE Global Communications Conference (GLOBECOM)*. IEEE, pp. 1–6.
- Braekers, Kris, Katrien Ramaekers, and Inneke Van Nieuwenhuysse (2016). "The vehicle routing problem: State of the art classification and review". In: *Computers & Industrial Engineering* 99, pp. 300–313.
- Brandão, José (2016). "A deterministic iterated local search algorithm for the vehicle routing problem with backhauls". In: *Top* 24.2, pp. 445–465.
- (2020). "A memory-based iterated local search algorithm for the multi-depot open vehicle routing problem". In: *European Journal of Operational Research* 284.2, pp. 559–571.
- Brandão, Julliany S, Thiago F Noronha, Mauricio GC Resende, and Celso C Ribeiro (2015). "A biased random-key genetic algorithm for single-round divisible load scheduling". In: *International Transactions in Operational Research* 22.5, pp. 823–839.
- (2017). "A biased random-key genetic algorithm for scheduling heterogeneous multi-round systems". In: *International Transactions in Operational Research* 24.5, pp. 1061–1077.
- Breunig, Ulrich, Roberto Baldacci, Richard F Hartl, and Thibaut Vidal (2019). "The electric two-echelon vehicle routing problem". In: *Computers & Operations Research* 103, pp. 198–210.
- Bruck, Bruno P, Valerio Incerti, Manuel Iori, and Matteo Vignoli (2017). "Minimizing CO₂ emissions in a practical daily carpooling problem". In: *Computers & Operations Research* 81, pp. 40–50.
- Bruck, Bruno P and Manuel Iori (2017). "Non-elementary formulations for single vehicle routing problems with pickups and deliveries". In: *Operations Research* 65.6, pp. 1597–1614.
- Brunner, Edgar, Holger Dette, and Axel Munk (1997). "Box-type approximations in nonparametric factorial designs". In: *Journal of the American Statistical Association* 92.440, pp. 1494–1502.
- Burke, Edmund K., Timothy Curtois, Gerhard Post, Rong Qu, and Bart Veltman (2008). "A hybrid heuristic ordering and variable neighbourhood search for the nurse rostering problem". In: *European Journal of Operational Research* 188.2, pp. 330–341.
- Cabrera, Guillem, Angel A Juan, Daniel Lázaro, Joan M Marquès, and Iuliia 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-Cruz, Jose, Pol Arias, Daniel Guimaraes, Daniel Riera, and Angel A Juan (2015). "Rich vehicle routing problem: Survey". In: *ACM Computing Surveys (CSUR)* 47.2, p. 32.
- Cai, Hua, Xi Wang, Peter Adriaens, and Ming Xu (2019). "Environmental benefits of taxi ride sharing in Beijing". In: *Energy* 174, pp. 503–508.
- Cakici, Eray, Scott J Mason, H Neil Geismar, and John W Fowler (2014). "Scheduling parallel machines with single vehicle delivery". In: *Journal of Heuristics* 20.5, pp. 511–537.
- Calvet, Laura, J sica de Armas, David Masip, and Angel A Juan (2017). "Learnheuristics: hybridizing metaheuristics with machine learning for optimization with dynamic inputs". In: *Open Mathematics* 15.1, pp. 261–280.
- Calvet, Laura, Albert Ferrer, M Isabel Gomes, Angel A Juan, and David Masip (2016a). "Combining statistical learning with metaheuristics for the multi-depot vehicle routing problem with market segmentation". In: *Computers & Industrial Engineering* 94, pp. 93–104.
- Calvet, Laura, Angel A Juan, Carles Serrat, and Jana Ries (2016b). "A statistical learning based approach for parameter fine-tuning of metaheuristics". In: *SORT-Statistics and Operations Research Transactions* 1.1, pp. 201–224.
- Caro, Felipe, Victor Mart nez-de Alb niz, and Paat Rusmevichientong (2014). "The assortment packing problem: Multiperiod assortment planning for short-lived products". In: *Management Science* 60.11, pp. 2701–2721.

- Çatay, Bülent (2010). "A new saving-based ant algorithm for the vehicle routing problem with simultaneous pickup and delivery". In: *Expert Systems with Applications* 37.10, pp. 6809–6817.
- Charikar, Moses and Sudipto Guha (1999). "Improved combinatorial algorithms for the facility location and k-median problems". In: *40th Annual Symposium on Foundations of Computer Science*. New York City, NY, USA, pp. 378–388.
- Cheikh-Graiet, Sondes Ben, Mariagrazia Dotoli, and Slim Hammadi (2020). "A Tabu Search based metaheuristic for dynamic carpooling optimization". In: *Computers & Industrial Engineering* 140, p. 106217.
- Chen, Dawei, Shuangli Pan, Qun Chen, and Jiahui Liu (2020). "Vehicle routing problem of contactless joint distribution service during COVID-19 pandemic". In: *Transportation Research Interdisciplinary Perspectives* 8, p. 100233.
- Chen, Wenyi, Martijn Mes, Marco Schutten, and Job Quint (2019). "A ride-sharing problem with meeting points and return restrictions". In: *Transportation Science* 53.2, pp. 401–426.
- Chen, Zhi-Long (2004). "Integrated production and distribution operations". In: *Handbook of quantitative supply chain analysis*. Springer, pp. 711–745.
- (2010). "Integrated production and outbound distribution scheduling: review and extensions". In: *Operations research* 58.1, pp. 130–148.
- Chen, Zhi-Long and George L Vairaktarakis (2005). "Integrated scheduling of production and distribution operations". In: *Management Science* 51.4, pp. 614–628.
- Cheng, Ba-Yi, Joseph Y-T Leung, and Kai Li (2015). "Integrated scheduling of production and distribution to minimize total cost using an improved ant colony optimization method". In: *Computers & Industrial Engineering* 83, pp. 217–225.
- Chica, Manuel, Angel A Juan Pérez, Oscar Cordon, and David Kelton (2017). "Why simheuristics? Benefits, limitations, and best practices when combining metaheuristics with simulation". In: *Benefits, Limitations, and Best Practices When Combining Metaheuristics with Simulation (January 1, 2017)*.
- Choi, Keunho, Donghee Yoo, Gunwoo Kim, and Yongmoo Suh (2012). "A hybrid online-product recommendation system: Combining implicit rating-based collaborative filtering and sequential pattern analysis". In: *Electronic Commerce Research and Applications* 11.4, pp. 309–317.
- Choi, Sang Hyun and Yoon Ho Cho (2004). "An utility range-based similar product recommendation algorithm for collaborative companies". In: *Expert Systems with Applications* 27.4, pp. 549–557.
- Choi, Sang Hyun, Sungmin Kang, and Young Jun Jeon (2006). "Personalized recommendation system based on product specification values". In: *Expert Systems with Applications* 31.3, pp. 607–616.
- Chopra, Sunil (2016). "How omni-channel can be the future of retailing". In: *Decision* 43.2, pp. 135–144.
- Chudak, Fabián A and David B Shmoys (2003). "Improved approximation algorithms for the uncapacitated facility location problem". In: *SIAM Journal on Computing* 33.1, pp. 1–25.
- Cici, Blerim, Athina Markopoulou, and Nikolaos Laoutaris (2015). "Designing an on-line ride-sharing system". In: *Proceedings of the 23rd SIGSPATIAL International Conference on Advances in Geographic Information Systems*, pp. 1–4.
- Clarke, Geoff and John W Wright (1964). "Scheduling of vehicles from a central depot to a number of delivery points". In: *Operations Research* 12.4, pp. 568–581.
- Cohen, Morris A and Suman Mallik (1997). "Global supply chains: research and applications". In: *Production and Operations Management* 6.3, pp. 193–210.
- Cohen, Rami, Liane Lewin-Eytan, Joseph Seffi Naor, and Danny Raz (2015). "Near optimal placement of virtual network functions". In: *2015 IEEE Conference on Computer Communications (INFOCOM)*. Kowloon, Hong Kong, pp. 1346–1354.

- Colorni, Alberto, Marco Dorigo, Francesco Maffioli, Vittorio Maniezzo, GIOVANNI Righini, and Marco Trubian (1996). "Heuristics from nature for hard combinatorial optimization problems". In: *International Transactions in Operational Research* 3.1, pp. 1–21.
- Condotta, Alessandro, Sigrid Knust, Dimitri Meier, and Natalia V Shakhlevich (2013). "Tabu search and lower bounds for a combined production–transportation problem". In: *Computers & Operations Research* 40.3, pp. 886–900.
- Cook, Deborah J, David L Sackett, and Walter O Spitzer (1995). "Methodologic guidelines for systematic reviews of randomized control trials in health care from the Potsdam Consultation on Meta-Analysis". In: *Journal of Clinical Epidemiology* 48.1, pp. 167–171.
- Corlu, Canan G, Rocio de la Torre, Adrian Serrano-Hernandez, Angel A Juan, and Javier Faulin (2020). "Optimizing Energy Consumption in Transportation: Literature Review, Insights, and Research Opportunities". In: *Energies* 13.5, p. 1115.
- Cornuéjols, Gérard, George Nemhauser, and Laurence Wolsey (1983). *The Uncapacitated Facility Location Problem*. Tech. rep. 605. Ithaca, NY, USA: Cornell University.
- Crainic, Teodor G and Gilbert Laporte (2012). *Fleet management and logistics*. Springer Science & Business Media.
- Crainic, Teodor Gabriel, Simona Mancini, Guido Perboli, and Roberto Tadei (2013). "GRASP with path relinking for the two-echelon vehicle routing problem". In: *Advances in Metaheuristics*. Springer, pp. 113–125.
- Crainic, Teodor Gabriel, Guido Perboli, Simona Mancini, and Roberto Tadei (2010). "Two-echelon vehicle routing problem: a satellite location analysis". In: *Procedia-Social and Behavioral Sciences* 2.3, pp. 5944–5955.
- Crainic, Teodor Gabriel, Nicoletta Ricciardi, and Giovanni Storch (2004). "Advanced freight transportation systems for congested urban areas". In: *Transportation Research Part C: Emerging Technologies* 12.2, pp. 119–137.
- Crispim, José and José Brandão (2005). "Metaheuristics applied to mixed and simultaneous extensions of vehicle routing problems with backhauls". In: *Journal of the Operational Research Society* 56.11, pp. 1296–1302.
- Cui, Ligang, Jie Deng, Rui Liu, Dongyang Xu, Yajun Zhang, and Maozeng Xu (2020). "A stochastic multi-item replenishment and delivery problem with lead-time reduction initiatives and the solving methodologies". In: *Applied Mathematics and Computation* 374, p. 125055.
- Daknama, Rami and Elisabeth Kraus (2017). "Vehicle routing with drones". In: *arXiv preprint arXiv:1705.06431*.
- Dantzig, George B and John H Ramser (1959). "The truck dispatching problem". In: *Management science* 6.1, pp. 80–91.
- De Armas, Jesica, Angel A Juan, Joan M Marquès, and João Pedro Pedroso (2017). "Solving the deterministic and stochastic uncapacitated facility location problem: from a heuristic to a simheuristic". In: *Journal of the Operational Research Society* 68.10, pp. 1161–1176.
- Deif, I and L Bodin (1984). "Extension of the Clarke and Wright algorithm for solving the vehicle routing problem with backhauling". In: *Proceedings of the Babson conference on software uses in transportation and logistics management*. Babson Park, MA, pp. 75–96.
- Del Ser, Javier, Ana I Torre-Bastida, Ibai Lana, Miren Nekane Bilbao, and Cristina Perfecto (2017). "Nature-inspired heuristics for the multiple-vehicle selective pickup and delivery problem under maximum profit and incentive fairness criteria". In: *2017 IEEE Congress on Evolutionary Computation (CEC)*. IEEE, pp. 480–487.
- Demir, Emrah, Tolga Bektas, and Gilbert Laporte (2012). "An adaptive large neighborhood search heuristic for the pollution-routing problem". In: *European Journal of Operational Research* 223.2, pp. 346–359.

- (2014). “A review of recent research on green road freight transportation”. In: *European Journal of Operational Research* 237.3, pp. 775–793.
- Diniz, Débora N, Marcone JF Souza, Claudia M Carneiro, Daniela M Ushizima, Fátima NS de Medeiros, Paulo HC Oliveira, and Andrea GC Bianchi (2019). “An Iterated Local Search-Based Algorithm to Support Cell Nuclei Detection in Pap Smears Test”. In: *International Conference on Enterprise Information Systems*. Springer, pp. 78–96.
- Do, Phan-Thuan, Ngoc-Quang Nguyen, Duc-Nghia Nguyen, et al. (2016). “A practical dynamic share-a-ride problem with speed windows for Tokyo city”. In: *2016 Eighth International Conference on Knowledge and Systems Engineering (KSE)*. IEEE, pp. 55–60.
- Dominguez, Oscar, Daniel Guimarães, Angel A Juan, and Ignacio 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* 255.2, pp. 442–462.
- Dominguez, Oscar, Angel A. Juan, and Javier Faulin (2014). “A biased-randomized algorithm for the two-dimensional vehicle routing problem with and without item rotations”. In: *International Transactions in Operational Research* 21.3, pp. 375–398.
- Dominguez, Oscar, Angel A Juan, Ignacio de La Nuez, and Djamila Ouelhadj (2016b). “An ILS-biased randomization algorithm for the two-dimensional loading HFVRP with sequential loading and items rotation”. In: *Journal of the Operational Research Society* 67.1, pp. 37–53.
- Dominguez, Oscar, Angel A Juan, Ignacio de la Nuez, and Djamila Ouelhadj (2016c). “An ILS-biased randomization algorithm for the two-dimensional loading HFVRP with sequential loading and items rotation”. In: *Journal of the Operational Research Society* 67.1, pp. 37–53.
- Dorling, Kevin, Jordan Heinrichs, Geoffrey G Messier, and Sebastian Magierowski (2017). “Vehicle routing problems for drone delivery”. In: *IEEE Transactions on Systems, Man, and Cybernetics: Systems* 47.1, pp. 70–85.
- Drexl, Michael (2012). “Synchronization in vehicle routing: a survey of VRPs with multiple synchronization constraints”. In: *Transportation Science* 46.3, pp. 297–316.
- Duan, Yubin, Turash Mosharraf, Jie Wu, and Huanyang Zheng (2018). “Optimizing carpool scheduling algorithm through partition merging”. In: *2018 IEEE International Conference on Communications (ICC)*. IEEE, pp. 1–6.
- Duarte, Abraham, Manuel Laguna, and Rafael Marti (2018). *Metaheuristics for Business Analytics*. Springer.
- Dumas, Yvan, Jacques Desrosiers, and Francois Soumis (1991). “The pickup and delivery problem with time windows”. In: *European Journal of Operational Research* 54.1, pp. 7–22.
- Efroymsen, MA and TL Ray (1966). “A branch-bound algorithm for plant location”. In: *Operations Research* 14.3, pp. 361–368.
- Elia, Valerio and Maria Grazia Gnoni (2015). “Designing an effective closed loop system for pallet management”. In: *International Journal of Production Economics* 170, pp. 730–740.
- Ellegaard, Chris and Thomas Ritter (2007). “Attractiveness in business markets: conceptualization and propositions”. In: *White paper*, pp. 1–10.
- Erengüç, Ş Selçuk, Natalie C Simpson, and Asoo J Vakharia (1999). “Integrated production/distribution planning in supply chains: An invited review”. In: *European Journal of Operational Research* 115.2, pp. 219–236.
- Erlenkotter, Donald (1978). “A dual-based procedure for uncapacitated facility location”. In: *Operations Research* 26.6, pp. 992–1009.
- Estrada-Moreno, Alejandro, Albert Ferrer, Angel A Juan, Adil Bagirov, and Javier Panadero (2019). “A biased-randomised algorithm for the capacitated facility location problem with soft constraints”. In: *Journal of the Operational Research Society* 0.0, pp. 1–17.

- European Commission (2016). *A European Strategy for low-emission mobility*. https://ec.europa.eu/clima/policies/transport_en, Last accessed on 2020-09-04.
- European Environment Agency (2019). *Greenhouse gas emissions from transport in Europe*. <https://www.eea.europa.eu/data-and-maps/indicators/transport-emissions-of-greenhouse-gases/transport-emissions-of-greenhouse-gases-12>, Last accessed on 2020-09-04.
- Ezzatneshan, Aziz (2010). "A algorithm for the Vehicle Problem". In: *International Journal of Advanced Robotic Systems* 7.2, p. 14.
- Fagnant, Daniel J. and Kara M. Kockelman (2015). "Dynamic Ride-Sharing and Optimal Fleet Sizing for a System of Shared Autonomous Vehicles". In: *Compendium of Transportation Research Board 94th Annual Meeting*.
- Falcon, Rafael, Xu Li, Amiya Nayak, and Ivan Stojmenovic (2010). "The one-commodity traveling salesman problem with selective pickup and delivery: An ant colony approach". In: *IEEE Congress on Evolutionary Computation*. IEEE, pp. 1–8.
- Fan, Li, Gangyan Li, Jian Yang, Hongzhi Sun, and Zhanghua Luo (2015). "The design of IOV overall structure and the research of IOV key technology in intelligent transportation system". In: *Proceedings of the 2015 International Power, Electronics and Materials Engineering Conference*. Dalian, China.
- Fan, Xiangxiang, Yeming Gong, Xianhao Xu, and Bipan Zou (2019). "Optimal decisions in reducing loss rate of returnable transport items". In: *Journal of cleaner production* 214, pp. 1050–1060.
- Farahani, Poorya, Martin Grunow, and H-O Günther (2012). "Integrated production and distribution planning for perishable food products". In: *Flexible services and manufacturing journal* 24.1, pp. 28–51.
- Farhan, J. and T. Donna Chen (2018). "Impact of ridesharing on operational efficiency of shared autonomous electric vehicle fleet". In: *Transportation Research Part C: Emerging Technologies*.
- Fathollahi-Fard, Amir Mohammad, Mostafa Hajiaghahi-Keshteli, and Seyedali Mirjalili (2020). "A set of efficient heuristics for a home healthcare problem". In: *Neural Computing and Applications* 32.10, pp. 6185–6205.
- Faulin, Javier, Miquel Gilibert, Angel A Juan, Xavier Vilajosana, and Ruben Ruiz (2008a). "SR-1: A simulation-based algorithm for the capacitated vehicle routing problem". In: *2008 Winter Simulation Conference*. IEEE, pp. 2708–2716.
- Faulin, Javier and Angel A Juan (2008). "The ALGACEA-1 method for the capacitated vehicle routing problem". In: *International Transactions in Operational Research* 15.5, pp. 599–621.
- Faulin, Javier, Angel A Juan, Carles Serrat, and Vicente Bargueno (2008b). "Predicting availability functions in time-dependent complex systems with SAEDS simulation algorithms". In: *Reliability Engineering & System Safety* 93.11, pp. 1761–1771.
- Feo, Thomas A and Mauricio GC Resende (1995). "Greedy randomized adaptive search procedures". In: *Journal of global optimization* 6.2, pp. 109–133.
- Fernandez, Stephanie Alvarez, Angel A Juan, J sica de Armas Adri n, Daniel Guerreiro e Silva, and Daniel Riera Terr n (2018). "Metaheuristics in telecommunication systems: network design, routing, and allocation problems". In: *IEEE Systems Journal* 12.4, pp. 3948–3957.
- Ferone, Daniele, Aljoscha Gruler, Paola Festa, and Angel A. Juan (2018). "Enhancing and extending the classical GRASP framework with biased randomisation and simulation". In: *Journal of the Operational Research Society* 70.8, pp. 1–14.
- Ferone, Daniele, Aljoscha Gruler, Paola Festa, and Angel A Juan (2019). "Enhancing and extending the classical GRASP framework with biased randomisation and simulation". In: *Journal of the Operational Research Society* 70.8, pp. 1362–1375.

- Ferone, Daniele, Sara Hatami, Eliana M González-Neira, Angel A Juan, and Paola Festa (2020). "A biased-randomized iterated local search for the distributed assembly permutation flow-shop problem". In: *International Transactions in Operational Research* 27.3, pp. 1368–1391.
- Ferreira, Kris and Joel Goh (2019). "Assortment rotation and the value of concealment". In: *Harvard Business School Technology & Operations Mgt. Unit Working Paper* 041.17.
- Ferrer, Alberto, Daniel Guimarans, Helena Ramalinho, and Angel A Juan (2016). "A BRILS meta-heuristic for non-smooth flow-shop problems with failure-risk costs". In: *Expert Systems with Applications* 44, pp. 177–186.
- Festa, Paola and Mauricio GC Resende (2009). "An annotated bibliography of GRASP–Part I: Algorithms". In: *International Transactions in Operational Research* 16.1, pp. 1–24.
- Fikar, Christian, Angel A Juan, Enoc Martinez, and Patrick Hirsch (2016). "A discrete-event driven metaheuristic for dynamic home service routing with synchronised trip sharing". In: *European Journal of Industrial Engineering* 10.3, pp. 323–340.
- Flamand, Tulay, Ahmed Ghoniem, and Bacel Maddah (2016). "Promoting impulse buying by allocating retail shelf space to grouped product categories". In: *Journal of the Operational Research Society* 67.7, pp. 953–969.
- Fontaras, Georgios, Nikiforos-Georgios Zacharof, and Biagio Ciuffo (2017). "Fuel consumption and CO₂ emissions from passenger cars in Europe–Laboratory versus real-world emissions". In: *Progress in Energy and Combustion Science* 60, pp. 97–131.
- Fu, Liang-Liang, Mohamed Ali Aloulou, and Chefi Triki (2017). "Integrated production scheduling and vehicle routing problem with job splitting and delivery time windows". In: *International Journal of Production Research* 55.20, pp. 5942–5957.
- Fukasawa, Ricardo, Humberto Longo, Jens Lysgaard, Marcus Poggi de Aragão, Marcelo Reis, Eduardo Uchoa, and Renato F Werneck (2006). "Robust branch-and-cut-and-price for the capacitated vehicle routing problem". In: *Mathematical programming* 106.3, pp. 491–511.
- Furuhata, Masabumi, Maged Dessouky, Fernando Ordóñez, Marc-Etienne Brunet, Xiaoqing Wang, and Sven Koenig (2013). "Ridesharing: The state-of-the-art and future directions". In: *Transportation Research Part B: Methodological* 57, pp. 28–46.
- Gajpal, Yuvraj and Prakash Abad (2009). "An ant colony system (ACS) for vehicle routing problem with simultaneous delivery and pickup". In: *Computers & Operations Research* 36.12, pp. 3215–3223.
- Galino, Santiago and Antonio Moreno (2014). "Integration of online and offline channels in retail: The impact of sharing reliable inventory availability information". In: *Management Science* 60.6, pp. 1434–1451.
- Gallino, Santiago and Antonio Moreno (2014). "Integration of online and offline channels in retail: The impact of sharing reliable inventory availability information". In: *Management Science* 60.6, pp. 1434–1451.
- Gao, Long, Douglas J. Thomas, and Michael B. Freimer (2014). "Optimal inventory control with retail pre-packs". In: *Production and Operations Management* 23.10, pp. 1761–1778.
- García-Nájera, Abel, John A Bullinaria, and Miguel A Gutiérrez-Andrade (2015). "An evolutionary approach for multi-objective vehicle routing problems with backhauls". In: *Computers & Industrial Engineering* 81, pp. 90–108.
- Gatteschi, Valentina, Fabrizio Lamberti, Gianluca Paravati, Andrea Sanna, Claudio Demartini, Alberto Lisanti, and Giorgio Venezia (2015). "New frontiers of delivery services using drones: A prototype system exploiting a quadcopter for autonomous drug shipments". In: *2015 IEEE 39th Annual Computer Software and Applications Conference*. Vol. 2. IEEE, pp. 920–927.
- Gavish, Bezalel and Stephen C Graves (1978). *The travelling salesman problem and related problems (Working Paper)*. Tech. rep. Massachusetts Institute of Technology, Operations Research Center.

- Geismar, H Neil, Milind Dawande, and Chelliah Sriskandarajah (2011). "Pool-point distribution of zero-inventory products". In: *Production and Operations Management* 20.5, pp. 737–753.
- Geismar, H Neil, Gilbert Laporte, Lei Lei, and Chelliah Sriskandarajah (2008). "The integrated production and transportation scheduling problem for a product with a short lifespan". In: *INFORMS Journal on Computing* 20.1, pp. 21–33.
- Gendreau, Michel, Gilbert Laporte, and Jean-Yves Potvin (2002). "Metaheuristics for the capacitated VRP". In: *The vehicle routing problem*. SIAM, pp. 129–154.
- Gendreau, Michel and Jean-Yves Potvin (2005). "Tabu search". In: *Search methodologies*. Springer, pp. 165–186.
- Gerla, Mario, Eun-Kyu Lee, Giovanni Pau, and Uichin Lee (2014). "Internet of vehicles: From intelligent grid to autonomous cars and vehicular clouds". In: *2014 IEEE world forum on internet of things (WF-IoT)*. IEEE, pp. 241–246.
- Ghiani, Gianpaolo, Francesca Guerriero, Gilbert Laporte, and Roberto Musmanno (2003). "Real-time vehicle routing: Solution concepts, algorithms and parallel computing strategies". In: *European Journal of Operational Research* 151.1, pp. 1–11.
- Ghilas, Veaceslav, Emrah Demir, and Tom Van Woensel (2016). "An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows and scheduled lines". In: *Computers & Operations Research* 72, pp. 12–30.
- Ghosh, Diptesh (2003). "Neighborhood search heuristics for the uncapacitated facility location problem". In: *European Journal of Operational Research* 150.1, pp. 150–162.
- Gilmore, Paul C and Ralph E Gomory (1961). "A linear programming approach to the cutting-stock problem". In: *Operations research* 9.6, pp. 849–859.
- Glock, Christoph H (2017). "Decision support models for managing returnable transport items in supply chains: A systematic literature review". In: *International Journal of Production Economics* 183, pp. 561–569.
- Glover, Fred W and Gary A Kochenberger (2006). *Handbook of Metaheuristics*. Vol. 57. Springer Science & Business Media.
- Goetschalckx, Marc and Charlotte Jacobs-Blecha (1989). "The vehicle routing problem with backhauls". In: *European Journal of Operational Research* 42.1, pp. 39–51.
- Goetschalckx, Marc, Carlos J Vidal, and Koray Dogan (2002). "Modeling and design of global logistics systems: A review of integrated strategic and tactical models and design algorithms". In: *European Journal of Operational Research* 143.1, pp. 1–18.
- Goksal, Fatma Pinar, Ismail Karaoglan, and Fulya Altiparmak (2013). "A hybrid discrete particle swarm optimization for vehicle routing problem with simultaneous pickup and delivery". In: *Computers & Industrial Engineering* 65.1, pp. 39–53.
- Golden, Bruce L, Larry Levy, and Rakesh Vohra (1987). "The orienteering problem". In: *Naval Research Logistics (NRL)* 34.3, pp. 307–318.
- Gonçalves, José Fernando and Mauricio GC Resende (2011). "Biased random-key genetic algorithms for combinatorial optimization". In: *Journal of Heuristics* 17.5, pp. 487–525.
- Gonzalez-Martin, Sergio, Angel A Juan, Daniel Riera, Quim Castella, Rodrigo Muñoz, and Alejandra Perez (2012). "Development and assessment of the SHARP and RandSHARP algorithms for the arc routing problem". In: *AI Communications* 25.2, pp. 173–189.
- Gonzalez-Martin, Sergio, Angel A Juan, Daniel Riera, Monica G Elizondo, and Juan J Ramos (2018). "A simheuristic algorithm for solving the arc routing problem with stochastic demands". In: *Journal of Simulation* 12.1, pp. 53–66.
- Gonzalez-Neira, Eliana Maria, Daniele Ferone, Sara Hatami, and Angel A Juan (2017). "A biased-randomized simheuristic for the distributed assembly permutation flowshop problem with stochastic processing times". In: *Simulation Modelling Practice and Theory* 79, pp. 23–36.

- Govindan, Kannan and Hamed Soleimani (2017). "A review of reverse logistics and closed-loop supply chains: a Journal of Cleaner Production focus". In: *Journal of Cleaner Production* 142, pp. 371–384.
- Grasas, Alex, Angel A Juan, Javier Faulin, Jesica de Armas, and Helena Ramalhinho (2017). "Biased randomization of heuristics using skewed probability distributions: a survey and some applications". In: *Computers & Industrial Engineering* 110, pp. 216–228.
- Grasas, Alex, Angel A Juan, and Helena R Lourenço (2016). "SimILS: a simulation-based extension of the iterated local search metaheuristic for stochastic combinatorial optimization". In: *Journal of Simulation* 10.1, pp. 69–77.
- Gribkovskaia, Irina, Gilbert Laporte, and Aliaksandr Shyshou (2008). "The single vehicle routing problem with deliveries and selective pickups". In: *Computers & Operations Research* 35.9, pp. 2908–2924.
- Gruler, Aljoscha, Christian Fikar, Angel A Juan, Patrick Hirsch, and C Contreras-Bolton (2017). "Supporting multi-depot and stochastic waste collection management in clustered urban areas via simulation–optimization". In: *Journal of simulation* 11.1, pp. 11–19.
- Gruler, Aljoscha, Javier Panadero, Jesica de Armas, Jose A Moreno, and Angel A Juan (2018). "Combining variable neighborhood search with simulation for the inventory routing problem with stochastic demands and stock-outs". In: *Computers & Industrial Engineering* 123, pp. 278–288.
- (2020). "A variable neighborhood search simheuristic for the multiperiod inventory routing problem with stochastic demands". In: *International Transactions in Operational Research* 27.1, pp. 314–335.
- Gurobi Optimization, LLC (2020). *Gurobi Optimizer Reference Manual*. <http://www.gurobi.com>.
- Gutiérrez-Jarpa, Gabriel, Guy Desaulniers, Gilbert Laporte, and Vladimir Marianov (2010). "A branch-and-price algorithm for the vehicle routing problem with deliveries, selective pickups and time windows". In: *European Journal of Operational Research* 206.2, pp. 341–349.
- Györgyi, Péter and Tamás Kis (2019). "A probabilistic approach to pickup and delivery problems with time window uncertainty". In: *European Journal of Operational Research* 274.3, pp. 909–923.
- Hajiaghaei-Keshteli, Mostafa, M Aminnayeri, and SMT Fatemi Ghomi (2014). "Integrated scheduling of production and rail transportation". In: *Computers & Industrial Engineering* 74, pp. 240–256.
- Hajiaghaei-Keshteli, Mostafa and MJASC Aminnayeri (2014). "Solving the integrated scheduling of production and rail transportation problem by Keshtel algorithm". In: *Applied Soft Computing* 25, pp. 184–203.
- Hakli, Huseyin and Zeynep Ortacay (2019). "An improved scatter search algorithm for the uncapacitated facility location problem". In: *Computers & Industrial Engineering* 135, pp. 855–867.
- Halinen, Aino (2012). *Relationship marketing in professional services: a study of agency-client dynamics in the advertising sector*. Routledge.
- Han, F., S. Zhao, L. Zhang, and J. Wu (2016). "Survey of Strategies for Switching Off Base Stations in Heterogeneous Networks for Greener 5G Systems". In: *IEEE Access* 4, pp. 4959–4973.
- Hanafi, Saïd, Renata Mansini, and Roberto Zanotti (2020). "The multi-visit team orienteering problem with precedence constraints". In: *European Journal of Operational Research* 282.2, pp. 515–529.
- Hansen, Pierre and Nenad Mladenović (2001). "Variable neighborhood search: Principles and applications". In: *European Journal of Operational Research* 130.3, pp. 449–467.
- (2003). "Variable neighborhood search". In: *Handbook of Metaheuristics*. Springer, pp. 145–184.
- (2014). "Variable neighborhood search". In: *Search methodologies*. Springer, pp. 313–337.
- Haouari, Mohamed and Lotfi Hidri (2008). "On the hybrid flowshop scheduling problem". In: *International Journal of Production Economics* 113.1, pp. 495–497.
- Hartley, James and Ronald N Kostoff (2003). "How useful are key words' in scientific journals?" In: *Journal of Information Science* 29.5, pp. 433–438.

- Hatami, Sara, Laura Calvet, Victor Fernández-Viagas, José M Framiñán, and Angel A Juan (2018). "A simheuristic algorithm to set up starting times in the stochastic parallel flowshop problem". In: *Simulation Modelling Practice and Theory* 86, pp. 55–71.
- He, Wen, Kai Hwang, and Deyi Li (2014). "Intelligent carpool routing for urban ridesharing by mining GPS trajectories". In: *IEEE Transactions on Intelligent Transportation Systems* 15.5, pp. 2286–2296.
- Heitz-Spahn, Sandrine (2013). "Cross-channel free-riding consumer behavior in a multichannel environment: An investigation of shopping motives, sociodemographics and product categories". In: *Journal of Retailing and Consumer Services* 20.6, pp. 570–578.
- Hellström, Daniel and Ola Johansson (2010). "The impact of control strategies on the management of returnable transport items". In: *Transportation Research Part E: Logistics and Transportation Review* 46.6, pp. 1128–1139.
- Hemmelmayr, Vera C, Jean-François Cordeau, and Teodor Gabriel Crainic (2012). "An adaptive large neighborhood search heuristic for two-echelon vehicle routing problems arising in city logistics". In: *Computers & operations research* 39.12, pp. 3215–3228.
- Henderson, Darrall, Sheldon H Jacobson, and Alan W Johnson (2003). "The theory and practice of simulated annealing". In: *Handbook of Metaheuristics*. Springer, pp. 287–319.
- Herbawi, Wesam and Michael Weber (2012). "The ridematching problem with time windows in dynamic ridesharing: A model and a genetic algorithm". In: *2012 IEEE Congress on Evolutionary Computation*. IEEE, pp. 1–8.
- Herrera, Juliver Gil and Juan Felipe Botero (2016). "Resource allocation in NFV: A comprehensive survey". In: *IEEE Transactions on Network and Service Management* 13.3, pp. 518–532.
- Hiermann, Gerhard, Matthias Prandtstetter, Andrea Rendl, Jakob Puchinger, and Günther R Raidl (2015). "Metaheuristics for solving a multimodal home-healthcare scheduling problem". In: *Central European Journal of Operations Research* 23.1, pp. 89–113.
- Ho, Sin C, WY Szeto, Yong-Hong Kuo, Janny MY Leung, Matthew Petering, and Terence WH Tou (2018). "A survey of dial-a-ride problems: Literature review and recent developments". In: *Transportation Research Part B: Methodological* 111, pp. 395–421.
- Hoffman, Karla L and Ted K Ralphs (2013). "Integer and combinatorial optimization". In: *Encyclopedia of Operations Research and Management Science*, pp. 771–783.
- Homayouni, S Mahdi, Dalila BMM Fontes, and Fernando ACC Fontes (2019). "A BRKGA for the integrated scheduling problem in FMSs". In: *Proceedings of the Genetic and Evolutionary Computation Conference Companion*, pp. 287–288.
- Homayouni, S Mahdi, Dalila BMM Fontes, and José F Gonçalves (2020). "A multistart biased random key genetic algorithm for the flexible job shop scheduling problem with transportation". In: *International Transactions in Operational Research*.
- Honhon, Dorothée, Vishal Gaur, and Sridhar Seshadri (2010). "Assortment Planning and Inventory Decisions Under Stockout-Based Substitution". In: *Operations Research* 58.5, pp. 1364–1379.
- Hosni, Hadi, Joe Naoum-Sawaya, and Hassan Artail (2014). "The shared-taxi problem: Formulation and solution methods". In: *Transportation Research Part B: Methodological* 70, pp. 303–318.
- Howard, Daniel and Danielle Dai (2014). "Public perceptions of self-driving cars: The case of Berkeley, California". In: *Transportation Research Board* 14.4502, pp. 1–16.
- Huang, Shih Chia, Ming Kai Jiau, and Chih Hsiang Lin (2015). "A genetic-algorithm-based approach to solve carpool service problems in cloud computing". In: *IEEE Transactions on Intelligent Transportation Systems* 16.1, pp. 352–364.
- Hübner, Alexander, Andreas Holzapfel, and Heinrich Kuhn (2016a). "Distribution systems in omnichannel retailing". In: *Business Research* 9.2, pp. 255–296.

- Hübner, Alexander, Johannes Wollenburg, and Andreas Holzzapfel (2016b). "Retail logistics in the transition from multi-channel to omni-channel". In: *International Journal of Physical Distribution & Logistics Management* 46.6/7, pp. 562–583.
- Hübner, Alexander H and Heinrich Kuhn (2012). "Retail category management: State-of-the-art review of quantitative research and software applications in assortment and shelf space management". In: *Omega* 40.2, pp. 199–209.
- Husakou, Anatol, Lars Magnus Hvattum, Ketil Danielsen, and Arild Hoff (2020). "An application of the multi-depot heterogeneous fixed fleet open vehicle routing problem". In: *International Journal of Advanced Operations Management* 12.2, pp. 142–155.
- Ilic, Alexander, Jason WP Ng, Paul Bowman, and Thorsten Staake (2009). "The value of RFID for RTI management". In: *Electronic Markets* 19.2-3, pp. 125–135.
- ISO (2016). *ISO/IEC 19762:2016. Information technology – Automatic identification and data capture (AIDC) techniques – Harmonized vocabulary.*
- Jacobs-Blecha, Charlotte and Marc Goetschalckx (1992). "The vehicle routing problem with back-hauls: properties and solution algorithms". In: *National Transportation Research Board* 13.
- Jacobsen, S Kruse and Oli BG Madsen (1980). "A comparative study of heuristics for a two-level routing-location problem". In: *European Journal of Operational Research* 5.6, pp. 378–387.
- Jain, Kamal, Mohammad Mahdian, and Amin Saberi (2002). "A new greedy approach for facility location problems". In: *Proceedings of the thirty-fourth annual ACM symposium on Theory of computing.* Montreal, Quebec, Canada, pp. 731–740.
- Jain, Kamal and Vijay V Vazirani (2001). "Approximation algorithms for metric facility location and k-median problems using the primal-dual schema and Lagrangian relaxation". In: *Journal of the ACM* 48.2, pp. 274–296.
- Ji, B., X. Zhang, S. Mumtaz, C. Han, C. Li, H. Wen, and D. Wang (2020). "Survey on the Internet of Vehicles: Network Architectures and Applications". In: *IEEE Communications Standards Magazine* 4.1, pp. 34–41.
- Johnson, B (2015). "Intermountain healthcare supply chain". In: *The 2015 Healthcare Supply Chain Conference, New Orleans, LA, February*, pp. 21–25.
- Juan, Angel A., Javier Faulin, Albert Ferrer, Helena R. Lourenço, and Barry Barrios (2013a). "MIRHA: multi-start biased randomization of heuristics with adaptive local search for solving non-smooth routing problems". In: *TOP* 21.1, pp. 109–132.
- Juan, Angel A, Javier Faulin, Scott E Grasman, Markus Rabe, and Gonçalo Figueira (2015a). "A review of simheuristics: Extending metaheuristics to deal with stochastic combinatorial optimization problems". In: *Operations Research Perspectives* 2.1, pp. 62–72.
- Juan, Angel A, Javier Faulin, Josep Jorba, Jose Caceres, and Joan Manuel Marquès (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, Angel A, Javier Faulín, Josep Jorba, Daniel Riera, David Masip, and Barry Barrios (2011). "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* 62.6, pp. 1085–1097.
- Juan, Angel A, Javier Faulin, Rubén Ruiz, Barry Barrios, and Santi Caballé (2010). "The SR-GCWS hybrid algorithm for solving the capacitated vehicle routing problem". In: *Applied Soft Computing* 10.1, pp. 215–224.
- Juan, Angel A, W David Kelton, Christine SM Currie, and Javier Faulin (2018). "Simheuristics applications: dealing with uncertainty in logistics, transportation, and other supply chain areas". In: *Proceedings of the 2018 Winter Simulation Conference.* IEEE, pp. 3048–3059.

- Juan, Angel A, Helena R Lourenço, Manuel Mateo, Rachel Luo, and Quim Castella (2014). "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, Angel A, Carlos Mendez, Javier Faulin, Jesica de Armas, and Scott Grasman (2016). "Electric vehicles in logistics and transportation: A survey on emerging environmental, strategic, and operational challenges". In: *Energies* 9.2, p. 86.
- Juan, Angel A, Iñaki Pascual, Daniel Guimarans, and Barry Barrios (2015b). "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.
- Jung, Jaeyoung, R. Jayakrishnan, and Ji Young Park (2016). "Dynamic Shared-Taxi Dispatch Algorithm with Hybrid-Simulated Annealing". In: *Computer-Aided Civil and Infrastructure Engineering* 31.4, pp. 275–291.
- Kaiwartya, Omprakash, Abdul Hanan Abdullah, Yue Cao, Ayman Altameem, Mukesh Prasad, Chinteng Lin, and Xiulei Liu (2016). "Internet of vehicles: Motivation, layered architecture, network model, challenges, and future aspects". In: *IEEE Access* 4, pp. 5356–5373.
- Kaminskas, Marius, Derek Bridge, Franclin Foping, and Donogh Roche (2017). "Product-Seeded and Basket-Seeded Recommendations for Small-Scale Retailers". In: *Journal on Data Semantics* 6.1, pp. 3–14.
- Kang, He-Yau, WL Pearn, I-Ping Chung, and Amy HI Lee (2016). "An enhanced model for the integrated production and transportation problem in a multiple vehicles environment". In: *Soft Computing* 20.4, pp. 1415–1435.
- Karaboga, Dervis, Beyza Gorkemli, Celal Ozturk, and Nurhan Karaboga (2014). "A comprehensive survey: artificial bee colony (ABC) algorithm and applications". In: *Artificial Intelligence Review* 42.1, pp. 21–57.
- Karabulut, Korhan and M Fatih Tasgetiren (2020). "An evolution strategy approach to the team orienteering problem with time windows". In: *Computers & Industrial Engineering* 139, p. 106109.
- Karaođlan, İsmail and Saadettin Erhan Kesen (2017). "The coordinated production and transportation scheduling problem with a time-sensitive product: a branch-and-cut algorithm". In: *International Journal of Production Research* 55.2, pp. 536–557.
- Kenney, J. B. (2011). "Dedicated Short-Range Communications (DSRC) Standards in the United States". In: *Proceedings of the IEEE* 99.7, pp. 1162–1182.
- Keskin, Merve and Bülent Çatay (2016). "Partial recharge strategies for the electric vehicle routing problem with time windows". In: *Transportation Research Part C: Emerging Technologies* 65, pp. 111–127.
- Khademi Zareh, Hassan, Shiva Ghaffari, Ahmad Sadeghieh, and Ali Mostafaeipour (2019). "Designing a ride-sharing transportation system for assignment and transfer of passengers to a common destination". In: *Journal of Industrial and Systems Engineering* 12.3, pp. 141–153.
- Khayal, Danya, Rojee Pradhananga, Shaligram Pokharel, and Fatih Mutlu (2015). "A model for planning locations of temporary distribution facilities for emergency response". In: *Socio-Economic Planning Sciences* 52, pp. 22–30.
- Khezrian, Amir, Terence D Todd, George Karakostas, and Morteza Azimifar (2014). "Energy-efficient scheduling in green vehicular infrastructure with multiple roadside units". In: *IEEE Transactions on Vehicular Technology* 64.5, pp. 1942–1957.
- Khumawala, Basheer M (1972). "An efficient branch and bound algorithm for the warehouse location problem". In: *Management science* 18.12, B–718.
- Kim, Taebok, Christoph H Glock, and Yongjang Kwon (2014). "A closed-loop supply chain for deteriorating products under stochastic container return times". In: *Omega* 43, pp. 30–40.

- Kizys, Renatas, Angel A Juan, Bartosz Sawik, and Laura Calvet (2019). "A Biased-Randomized Iterated Local Search Algorithm for Rich Portfolio Optimization". In: *Applied Sciences* 9.17, p. 3509.
- Küçüköğlü, Ilker and Nursel Öztürk (2015). "An advanced hybrid meta-heuristic algorithm for the vehicle routing problem with backhauls and time windows". In: *Computers & Industrial Engineering* 86, pp. 60–68.
- Koç, Çağrı and Gilbert Laporte (2018). "Vehicle routing with backhauls: Review and research perspectives". In: *Computers & Operations Research* 91, pp. 79–91.
- Koç, Utku, Ayşegül Toptal, and Ihsan Sabuncuoğlu (2017). "Coordination of inbound and outbound transportation schedules with the production schedule". In: *Computers & Industrial Engineering* 103, pp. 178–192.
- Kök, A Gürhan, Marshall L Fisher, and Ramnath Vaidyanathan (2008). "Assortment planning: Review of literature and industry practice". In: *Retail supply chain management*. Springer, pp. 99–153.
- Kong, Min, Xinbao Liu, Jun Pei, Hao Cheng, and Panos M Pardalos (2020). "A BRKGA-DE algorithm for parallel-batching scheduling with deterioration and learning effects on parallel machines under preventive maintenance consideration". In: *Annals of Mathematics and Artificial Intelligence* 88.1, pp. 237–267.
- Kornhauser, Alain L, Paul Mottola, and Brian Stephenson (1977). "Transportation efficiency and the feasibility of dynamic ride sharing". In: *Transportation Research Record* 1.650, pp. 43–48.
- Kroon, Leo and Gaby Vrijens (1995). "Returnable containers: an example of reverse logistics". In: *International Journal of Physical Distribution & Logistics Management* 25.2, pp. 56–68.
- Lahyani, Rahma, Anne-Lise Gouguenheim, and Leandro C Coelho (2019). "A hybrid adaptive large neighbourhood search for multi-depot open vehicle routing problems". In: *International Journal of Production Research* 57.22, pp. 6963–6976.
- Lam, Chiou-Peng, Martin Masek, Luke Kelly, Michael Papisimeon, and Lyndon Benke (2019). "A simheuristic approach for evolving agent behaviour in the exploration for novel combat tactics". In: *Operations Research Perspectives* 6, p. 100123.
- Landry, Sylvain and Martin Beaulieu (2013). "The challenges of hospital supply chain management, from central stores to nursing units". In: *Handbook of healthcare operations management*. Springer, pp. 465–482.
- Lapinskaitė, Indrė and Justina Kuckailytė (2014). "The impact of supply chain cost on the price of the final product". In: *Business, Management and Education* 12.1, pp. 109–126.
- Laporte, Gilbert, Hélène Mercure, and Yves Nobert (1986). "An exact algorithm for the asymmetrical capacitated vehicle routing problem". In: *Networks* 16.1, pp. 33–46.
- Law, Averill M and W David Kelton (2000). *Simulation Modeling and Analysis*. Vol. 3. McGraw-Hill New York.
- Lee, Alan and Martin Savelsbergh (2015). "Dynamic ridesharing: Is there a role for dedicated drivers?" In: *Transportation Research Part B: Methodological* 81, pp. 483–497.
- Lema, M. A., A. Laya, T. Mahmoodi, M. Cuevas, J. Sachs, J. Markendahl, and M. Dohler (2017). "Business Case and Technology Analysis for 5G Low Latency Applications". In: *IEEE Access* 5, pp. 5917–5935.
- Lenstra, J. K. and A. H. G. Rinnooy Kan (1981a). "Complexity of vehicle routing and scheduling problems". In: *Networks* 11.2, pp. 221–227.
- Lenstra, Jan Karel and AHG Rinnooy Kan (1981b). "Complexity of vehicle routing and scheduling problems". In: *Networks* 11.2, pp. 221–227.
- Levin, Michael W, Kara M Kockelman, Stephen D Boyles, and Tianxin Li (2017). "A general framework for modeling shared autonomous vehicles with dynamic network-loading and dynamic ride-sharing application". In: *Computers, Environment and Urban Systems* 64, pp. 373–383.

- Li, Baoxiang, Dmitry Krushinsky, Hajo A. Reijers, and Tom Van Woensel (2014a). "The Share-A-Ride Problem: People and parcels sharing taxis". In: *European Journal of Operational Research* 238.1, pp. 31–40.
- Li, Chung-Lun and George Vairaktarakis (2007). "Coordinating production and distribution of jobs with bundling operations". In: *IIE transactions* 39.2, pp. 203–215.
- Li, Chung-Lun, George Vairaktarakis, and Chung-Yee Lee (2005). "Machine scheduling with deliveries to multiple customer locations". In: *European Journal of Operational Research* 164.1, pp. 39–51.
- Li, Feiyue, Bruce Golden, and Edward Wasil (2007). "The open vehicle routing problem: Algorithms, large-scale test problems, and computational results". In: *Computers & operations research* 34.10, pp. 2918–2930.
- Li, Haibing and Andrew Lim (2003). "A metaheuristic for the pickup and delivery problem with time windows". In: *International Journal on Artificial Intelligence Tools* 12.02, pp. 173–186.
- Li, Ruimin, Zhiyong Liu, and Ruibo Zhang (2018a). "Studying the benefits of carpooling in an urban area using automatic vehicle identification data". In: *Transportation Research Part C: Emerging Technologies* 93, pp. 367–380.
- Li, Xin, Sangen Hu, Wenbo Fan, and Kai Deng (2018b). "Modeling an enhanced ridesharing system with meet points and time windows". In: *PloS one* 13.5.
- Li, Yinglei and Sung Hoon Chung (2020). "Ride-Sharing under Travel Time Uncertainty: Robust Optimization and Clustering Approaches". In: *Computers & Industrial Engineering*, p. 106601.
- Li, Yuan, Haoxun Chen, and Christian Prins (2016). "Adaptive large neighborhood search for the pickup and delivery problem with time windows, profits, and reserved requests". In: *European Journal of Operational Research* 252.1, pp. 27–38.
- Li, Zhaolin (Erick), Qiang Lu, and Masoud Talebian (2014b). "Online versus bricks-and-mortar retailing: a comparison of price, assortment and delivery time". In: *International Journal of Production Research* 53.13, pp. 1–14.
- Liao, Ching-Jong, Evi Tjandradjaja, and Tsui-Ping Chung (2012). "An approach using particle swarm optimization and bottleneck heuristic to solve hybrid flow shop scheduling problem". In: *Applied Soft Computing* 12.6, pp. 1755–1764.
- Lin, Shih-Wei, Zne-Jung Lee, Kuo-Ching Ying, and Chou-Yuan Lee (2009). "Applying hybrid metaheuristics for capacitated vehicle routing problem". In: *Expert Systems with Applications* 36.2, pp. 1505–1512.
- Lin, Sifeng, Jonathan F Bard, Ahmad I Jarrah, Xinhui Zhang, and Luis J Novoa (2017). "Route design for last-in, first-out deliveries with backhauling". In: *Transportation Research Part C: Emerging Technologies* 76, pp. 90–117.
- Liu, Duen Ren and Ya Yueh Shih (2005). "Hybrid approaches to product recommendation based on customer lifetime value and purchase preferences". In: *Journal of Systems and Software* 77.2, pp. 181–191.
- Lokhandwala, Mustafa and Hua Cai (2018). "Dynamic ride sharing using traditional taxis and shared autonomous taxis: A case study of NYC". In: *Transportation Research Part C: Emerging Technologies* 97, pp. 45–60.
- Londoño, Julio C, Rafael D Tordecilla, Leandro do C Martins, and Angel A Juan (2020). "A biased-randomized iterated local search for the vehicle routing problem with optional backhauls". In: *TOP*, pp. 1–30.
- Long, Jiancheng, Weimin Tan, WY Szeto, and Yao Li (2018). "Ride-sharing with travel time uncertainty". In: *Transportation Research Part B: Methodological* 118, pp. 143–171.

- Lopes, Thiago Cantos, Adalberto Sato Michels, Ricardo Lüders, and Leandro Magatão (2020). "A simheuristic approach for throughput maximization of asynchronous buffered stochastic mixed-model assembly lines". In: *Computers & Operations Research* 115, p. 104863.
- Lourenço, Helena R, Olivier C Martin, and Thomas Stützle (2003). "Iterated local search". In: *Handbook of Metaheuristics*. Springer, pp. 320–353.
- (2010). "Iterated local search: Framework and applications". In: *Handbook of Metaheuristics*. Springer, pp. 363–397.
- Lourenço, Helena R, José P Paixão, and Rita Portugal (2001). "Multiobjective metaheuristics for the bus driver scheduling problem". In: *Transportation science* 35.3, pp. 331–343.
- Low, Chinyao, Chien-Min Chang, Rong-Kwei Li, and Chia-Ling Huang (2014). "Coordination of production scheduling and delivery problems with heterogeneous fleet". In: *International Journal of Production Economics* 153, pp. 139–148.
- Lu, J., Z. Zhang, T. Hu, P. Yi, and J. Lan (2019). "A Survey of Controller Placement Problem in Software-Defined Networking". In: *IEEE Access* 7, pp. 24290–24307.
- Lu, Quan and Maged Dessouky (2004). "An exact algorithm for the multiple vehicle pickup and delivery problem". In: *Transportation Science* 38.4, pp. 503–514.
- Ma, Changxi, Ruichun He, and Wei Zhang (2018). "Path optimization of taxi carpooling". In: *PLoS One* 13.8, e0203221.
- Ma, Shaohui, Pingping Gao, and Hui Tan (2017). "The impact of subsidies and charging facilities on demand for electric vehicles in China". In: *Environment and Urbanization ASIA* 8.2, pp. 230–242.
- Ma, Shuo, Yu Zheng, and Ouri Wolfson (2013). "T-share: A large-scale dynamic taxi ridesharing service". In: *2013 IEEE 29th International Conference on Data Engineering (ICDE)*. IEEE, pp. 410–421.
- Macal, Charles M (2016). "Everything you need to know about agent-based modelling and simulation". In: *Journal of Simulation* 10.2, pp. 144–156.
- Macrina, Giusy, Luigi Di Puglia Pugliese, Francesca Guerriero, and Gilbert Laporte (2019). "The green mixed fleet vehicle routing problem with partial battery recharging and time windows". In: *Computers & Operations Research* 101, pp. 183–199.
- Mahajan, Siddharth and Garrett van Ryzin (2001). "Stocking Retail Assortments Under Dynamic Consumer Substitution". In: *Operations Research* 49.3, pp. 334–351.
- Mahmoudi, Monirehalsadat and Irandokht Parviziomran (2020). "Reusable packaging in supply chains: A review of environmental and economic impacts, logistics system designs, and operations management". In: *International Journal of Production Economics*, p. 107730.
- Malapert, Arnaud, Jean-Charles Régim, and Mohamed Rezugui (2016). "Embarrassingly parallel search in constraint programming". In: *Journal of Artificial Intelligence Research* 57, pp. 421–464.
- Mantrala, Murali K., Michael Levy, Barbara E. Kahn, Edward J. Fox, Peter Gaidarev, Bill Dankworth, and Denish Shah (2009). "Why is Assortment Planning so Difficult for Retailers? A Framework and Research Agenda". In: *Journal of Retailing* 85.1, pp. 71–83.
- Marmol, Mage, Leandro do C Martins, Sara Hatami, Angel A Juan, and Vicenc Fernandez (2020). "Using biased-randomized algorithms for the multi-period product display problem with dynamic attractiveness". In: *Algorithms* 13.2, p. 34.
- Martí, Rafael, Jose A. Lozano, Alexander Mendiburu, and Leticia Hernando (2016). "Multi-start Methods". In: *Handbook of Heuristics*. Ed. by Rafael Martí, Pardalos Panos, and Mauricio G.C. Resende. Cham: Springer International Publishing, pp. 1–21.
- Martí, Rafael, Mauricio GC Resende, and Celso C Ribeiro (2013). "Multi-start methods for combinatorial optimization". In: *European Journal of Operational Research* 226.1, pp. 1–8.
- Martin, Simon, Djamila Ouelhadj, Patrick Beullens, Ender Ozcan, Angel A Juan, and Edmund K Burke (2016). "A multi-agent based cooperative approach to scheduling and routing". In: *European Journal of Operational Research* 254.1, pp. 169–178.

- Martins, Leandro do C, Christopher Bayliss, Angel A Juan, Javier Panadero, and Mage Marmol (2020a). "A savings-based heuristic for solving the omnichannel vehicle routing problem with pick-up and delivery". In: *Transportation Research Procedia* 47, pp. 83–90.
- Martins, Leandro do C, Canan G Corlu, Angel A Juan, Mohamed A Masmoudi, et al. (2020b). "Optimizing Ride-Sharing Operations in Smart Sustainable Cities: Challenges and the Need for Agile Algorithms". In: *Computers & Industrial Engineering*, p. 107080.
- Martins, Leandro do C, Patrick Hirsch, and Angel A Juan (2020c). "Agile optimization of a two-echelon vehicle routing problem with pickup and delivery". In: *International Transactions in Operational Research* 28.1, pp. 201–221.
- Martins, Simone L and Celso C Ribeiro (2006). "Metaheuristics and applications to optimization problems in telecommunications". In: *Handbook of optimization in telecommunications*. Springer, pp. 103–128.
- Masmoudi, Mohamed Amine, Manar Hosny, Emrah Demir, and Naoufel Cheikhrouhou (2018). "A study on the heterogeneous fleet of alternative fuel vehicles: Reducing CO₂ emissions by means of biodiesel fuel". In: *Transportation Research Part D: Transport and Environment* 63, pp. 137–155.
- Masmoudi, Mohamed Amine, Manar Hosny, Emrah Demir, and Erwin Pesch (2020). "Hybrid adaptive large neighborhood search algorithm for the mixed fleet heterogeneous dial-a-ride problem". In: *Journal of Heuristics* 26.1, pp. 83–118.
- Mason, Alex, Andy Shaw, and Ahmed Al-Shamma'a (2012). "Peer-to-peer inventory management of returnable transport items: A design science approach". In: *Computers in Industry* 63.3, pp. 265–274.
- Masoud, Neda and R Jayakrishnan (2017). "A real-time algorithm to solve the peer-to-peer ride-matching problem in a flexible ridesharing system". In: *Transportation Research Part B: Methodological* 106, pp. 218–236.
- Mazza, D., A. Pagès-Bernaus, D. Tarchi, A. A. Juan, and G. E. Corazza (2018). "Supporting Mobile Cloud Computing in Smart Cities via Randomized Algorithms". In: *IEEE Systems Journal* 12.2, pp. 1598–1609.
- Mazza, D., D. Tarchi, and G. E. Corazza (2014). "A user-satisfaction based offloading technique for smart city applications". In: *2014 IEEE Global Communications Conference*. Austin, TX, USA, pp. 2783–2788.
- McConnan, Isobel (2004). *Humanitarian charter and minimum standards in disaster response*. Sphere Project.
- Mehmood, Yasir, Farhan Ahmad, Ibrar Yaqoob, Asma Adnane, Muhammad Imran, and Sghaier Guizani (2017). "Internet-of-things-based smart cities: Recent advances and challenges". In: *IEEE Communications Magazine* 55.9, pp. 16–24.
- Mehra, Amit, Subodha Kumar, and Jagmohan S Raju (2013). "Showrooming and the competition between store and online retailers". In: *Available at SSRN* 2200420.
- Meixell, Mary J and Vidyaranya B Gargeya (2005). "Global supply chain design: A literature review and critique". In: *Transportation Research Part E: Logistics and Transportation Review* 41.6, pp. 531–550.
- Mena, Carlos, Michael Bourlakis, Alexander Hübner, Johannes Wollenburg, and Andreas Holzapfel (2016). "Retail logistics in the transition from multi-channel to omni-channel". In: *International Journal of Physical Distribution & Logistics Management*.
- Mentzer, John T, William DeWitt, James S Keebler, Soonhong Min, Nancy W Nix, Carlo D Smith, and Zach G Zacharia (2001). "Defining supply chain management". In: *Journal of Business logistics* 22.2, pp. 1–25.
- Migdalas, Athanasios, Panos M Pardalos, and Sverre Storøy (2013). *Parallel computing in optimization*. Vol. 7. Springer Science & Business Media.

- Min, Hokey (1989). "The multiple vehicle routing problem with simultaneous delivery and pick-up points". In: *Transportation Research Part A: General* 23.5, pp. 377–386.
- Mladenović, Nenad and Pierre Hansen (1997). "Variable neighborhood search". In: *Computers & operations research* 24.11, pp. 1097–1100.
- Moeini, Mahdi, Zied Jemai, and Evren Sahin (2015). "Location and relocation problems in the context of the emergency medical service systems: a case study". In: *Central European Journal of Operations Research* 23.3, pp. 641–658.
- Montane, Fermin Alfredo Tang and Roberto Dieguez Galvao (2006). "A tabu search algorithm for the vehicle routing problem with simultaneous pick-up and delivery service". In: *Computers & Operations Research* 33, pp. 595–619.
- Moons, Karen, Geert Waeyenbergh, and Liliane Pintelon (2019). "Measuring the logistics performance of internal hospital supply chains—a literature study". In: *Omega* 82, pp. 205–217.
- Moons, Stef, Katrien Ramaekers, An Caris, and Yasemin Arda (2017). "Integrating production scheduling and vehicle routing decisions at the operational decision level: a review and discussion". In: *Computers & Industrial Engineering* 104, pp. 224–245.
- Mosquera, Ana, Cristina Olarte Pascual, and Emma Juaneda Ayensa (2017). "Understanding the customer experience in the age of omni-channel shopping". In: *Revista ICONO14 Revista científica de Comunicación y Tecnologías emergentes* 15.2, p. 92.
- Mou, Shandong, David J. Robb, and Nicole DeHoratius (2017). "Retail store operations: Literature review and research directions". In: *European Journal of Operational Research*.
- Mourad, Abood, Jakob Puchinger, and Chengbin Chu (2019). "A survey of models and algorithms for optimizing shared mobility". In: *Transportation Research Part B: Methodological*.
- Mu, Dong, Chao Wang, Fu Zhao, and John W. Sutherland (2016a). "Solving vehicle routing problem with simultaneous pickup and delivery using parallel simulated annealing algorithm". In: *International Journal of Shipping and Transport Logistics* 8, pp. 81–106.
- Mu, Dong, Chao Wang, Fu Zhao, and John W Sutherland (2016b). "Solving vehicle routing problem with simultaneous pickup and delivery using parallel simulated annealing algorithm". In: *International Journal of Shipping and Transport Logistics* 8.1, pp. 81–106.
- Naderi, B., Rubén Ruiz, and M. Zandieh (2010). "Algorithms for a realistic variant of flowshop scheduling". In: *Computers & Operations Research* 37.2, pp. 236–246.
- Nadzadeh, Ali and Behzad Kafash (2019). "Fuzzy capacitated location-routing problem with simultaneous pickup and delivery demands". In: *Transportation Letters* 11.1, pp. 1–19.
- Najmi, Ali, David Rey, and Taha H Rashidi (2017). "Novel dynamic formulations for real-time ride-sharing systems". In: *Transportation Research part E: Logistics and Transportation Review* 108, pp. 122–140.
- Nakib, Amir and El-Ghazali Talbi (2017). *Metaheuristics for medicine and biology*. Vol. 704. Springer.
- Nanry, William P and J Wesley Barnes (2000). "Solving the pickup and delivery problem with time windows using reactive tabu search". In: *Transportation Research Part B: Methodological* 34.2, pp. 107–121.
- Naoum-Sawaya, Joe, Randy Cogill, Bissan Ghaddar, Shravan Sajja, Robert Shorten, Nicole Taheri, Pierpaolo Tommasi, Rudi Verago, and Fabian Wirth (2015). "Stochastic optimization approach for the car placement problem in ridesharing systems". In: *Transportation Research Part B: Methodological* 80, pp. 173–184.
- Narayanan, Santhanakrishnan, Emmanouil Chaniotakis, and Constantinos Antoniou (2020). "Shared autonomous vehicle services: A comprehensive review". In: *Transportation Research Part C: Emerging Technologies* 111, pp. 255–293.
- Nawaz, Muhammad, E Emory Enscore, and Inyong Ham (1983). "A heuristic algorithm for the m-machine, n-job flow-shop sequencing problem". In: *Omega* 11.1, pp. 91–95.

- Nazif, Habibeh and Lai Soon Lee (2012). "Optimised crossover genetic algorithm for capacitated vehicle routing problem". In: *Applied Mathematical Modelling* 36.5, pp. 2110–2117.
- Nechita, Elena, Gloria-Cerasela Crişan, Sergiu-Mădălin Obreja, and Constantin-Sebastian Damian (2016). "Intelligent Carpooling System". In: *New Approaches in Intelligent Control*. Springer, pp. 43–72.
- Newbert, Scott L (2007). "Empirical research on the resource-based view of the firm: an assessment and suggestions for future research". In: *Strategic Management Journal* 28.2, pp. 121–146.
- Ni, Yuanzhi, Jianping He, Lin Cai, Jianping Pan, and Yuming Bo (2018). "Joint roadside unit deployment and service task assignment for Internet of Vehicles (IoV)". In: *IEEE Internet of Things Journal* 6.2, pp. 3271–3283.
- Nucci, Francesco (2021). "Multi-shift single-vehicle routing problem under fuzzy uncertainty". In: *Intelligent and Fuzzy Techniques: Smart and Innovative Solutions*. Springer, pp. 1620–1627.
- Öncan, Temel, Kuban Altınel, and Gilbert Laporte (2009). "A comparative analysis of several asymmetric traveling salesman problem formulations". In: *Computers & Operations Research* 36.3, pp. 637–654.
- Onggo, Bhakti Stephan, Javier Panadero, Canan G Corlu, and Angel A Juan (2019). "Agri-food supply chains with stochastic demands: A multi-period inventory routing problem with perishable products". In: *Simulation Modelling Practice and Theory* 97, p. 101970.
- Pacheco, Joaquín and Manuel Laguna (2020). "Vehicle routing for the urgent delivery of face shields during the COVID-19 pandemic". In: *Journal of Heuristics* 26.5, pp. 619–635.
- Pages-Bernaus, Adela, Helena Ramalinho, Angel A Juan, and Laura Calvet (2019). "Designing e-commerce supply chains: a stochastic facility–location approach". In: *International Transactions in Operational Research* 26.2, pp. 507–528.
- Pan, Quan-Ke, Ling Wang, Jun-Qing Li, and Jun-Hua Duan (2014). "A novel discrete artificial bee colony algorithm for the hybrid flowshop scheduling problem with makespan minimisation". In: *Omega* 45.Supplement C, pp. 42–56.
- Panadero, J, C Currie, AA Juan, and C Bayliss (2020a). "Maximizing reward from a team of surveillance drones under uncertainty conditions: a simheuristic approach". In: *European Journal of Industrial Engineering* 14, pp. 1–23.
- Panadero, Javier, Jana Doering, Renatas Kizys, Angel A Juan, and Angels Fito (2018). "A variable neighborhood search simheuristic for project portfolio selection under uncertainty". In: *Journal of Heuristics*, pp. 1–23.
- (2020b). "A variable neighborhood search simheuristic for project portfolio selection under uncertainty". In: *Journal of Heuristics* 26.3, pp. 353–375.
- Panadero, Javier, Angel A Juan, Christopher Bayliss, and Christine Currie (2020c). "Maximising reward from a team of surveillance drones: a simheuristic approach to the stochastic team orienteering problem". In: *European Journal of Industrial Engineering* 14.4, pp. 485–516.
- Pankratz, Giselher (2005). "A grouping genetic algorithm for the pickup and delivery problem with time windows". In: *OR Spectrum* 27.1, pp. 21–41.
- Papadimitriou, Christos H and Kenneth Steiglitz (1998). *Combinatorial optimization: algorithms and complexity*. Courier Corporation.
- Parhami, Behrooz (2006). *Introduction to parallel processing: algorithms and architectures*. Springer Science & Business Media.
- Parragh, Sophie N, Karl F Doerner, and Richard F Hartl (2008). "A survey on pickup and delivery problems". In: *Journal für Betriebswirtschaft* 58.1, pp. 21–51.
- Parsons, Andrew G (2011). "Atmosphere in fashion stores: do you need to change?" In: *Journal of fashion marketing and management: An international journal* 15.4, pp. 428–445.

- Paternina-Arboleda, Carlos D., Jairo R. Montoya-Torres, Milton J. Acero-Dominguez, and Maria C. Herrera-Hernandez (2008). "Scheduling jobs on a k-stage flexible flow-shop". In: *Annals of Operations Research* 164.1, pp. 29–40.
- Pentico, David W (2008). "The assortment problem: A survey". In: *European Journal of Operational Research* 190.2, pp. 295–309.
- Perdana, Tomy, Diah Chaerani, Audi Luqmanul Hakim Achmad, and Fernianda Rahayu Hermiatin (2020). "Scenarios for handling the impact of COVID-19 based on food supply network through regional food hubs under uncertainty". In: *Heliyon* 6.10, e05128.
- Pisinger, David and Stefan Ropke (2007). "A general heuristic for vehicle routing problems". In: *Computers & operations research* 34.8, pp. 2403–2435.
- Poikonen, Stefan, Xingyin Wang, and Bruce Golden (2017). "The vehicle routing problem with drones: Extended models and connections". In: *Networks* 70.1, pp. 34–43.
- Prieto, Marc, George Baltas, and Valentina Stan (2017). "Car sharing adoption intention in urban areas: what are the key sociodemographic drivers?" In: *Transportation Research Part A: Policy and Practice* 101, pp. 218–227.
- Quick, Jonathan D, Hans V Hogerzeil, James R Rankin, Maurice Nelson Graham Dukes, Richard Laing, Andrew Garnett, Ronald W O'Connor, et al. (1997). *Managing drug supply: the selection, procurement, distribution, and use of pharmaceuticals*. West Hartford, Connecticut: Kumarian Press.
- Quintero-Araujo, Carlos L, Juan Pablo Caballero-Villalobos, Angel A Juan, and Jairo R Montoya-Torres (2017). "A biased-randomized metaheuristic for the capacitated location routing problem". In: *International Transactions in Operational Research* 24.5, pp. 1079–1098.
- Quintero-Araujo, Carlos L., Aljoscha Gruler, Angel A. Juan, and Javier Faulin (2019). "Using horizontal cooperation concepts in integrated routing and facility-location decisions". In: *International Transactions in Operational Research* 26.2, pp. 551–576.
- Raba, David, Alejandro Estrada-Moreno, Javier Panadero, and Angel A Juan (2020). "A reactive simheuristic using online data for a real-life inventory routing problem with stochastic demands". In: *International Transactions in Operational Research* 27.6, pp. 2785–2816.
- Rabe, Markus, Maik Deininger, and Angel A Juan (2020). "Speeding up computational times in simheuristics combining genetic algorithms with discrete-Event simulation". In: *Simulation Modelling Practice and Theory* 103, p. 102089.
- Rajamma, Rajasree K., Audhesh K. Paswan, and Gopala Ganesh (2007). "Services purchased at brick and mortar versus online stores, and shopping motivation". In: *Journal of Services Marketing* 21.3, pp. 200–212.
- Reil, Sebastian, Andreas Bortfeldt, and Lars Mönch (2018). "Heuristics for vehicle routing problems with backhauls, time windows, and 3D loading constraints". In: *European Journal of Operational Research* 266.3, pp. 877–894.
- Resende, Mauricio GC and Celso C Ribeiro (2003). "Greedy randomized adaptive search procedures". In: *Handbook of Metaheuristics*. Springer, pp. 219–249.
- (2016). *Optimization by GRASP*. Springer.
- Resende, Mauricio GC and Renato F Werneck (2006). "A hybrid multistart heuristic for the uncapacitated facility location problem". In: *European Journal of Operational Research* 174.1, pp. 54–68.
- Rey, Antón, Manuel Prieto, José Ignacio Gómez, Christian Tenllado, and J Ignacio Hidalgo (2018). "A CPU-GPU parallel ant colony optimization solver for the vehicle routing problem". In: *International Conference on the Applications of Evolutionary Computation*. Springer, pp. 653–667.
- Reyes-Rubiano, L, AA Juan, Christopher Bayliss, J Panadero, J Faulin, and P Copado (2020). "A Biased-Randomized Learnheuristic for Solving the Team Orienteering Problem with Dynamic Rewards". In: *Transportation Research Procedia* 47, pp. 680–687.

- Rodríguez Bolívar, Manuel Pedro, Laura Alcaide Muñoz, Antonio M López Hernández, et al. (2010). "Trends of e-government research: Contextualization and research opportunities". In: *The International Journal of Digital Accounting Research* 10, pp. 87–111.
- Ropke, Stefan and Jean-François Cordeau (2009). "Branch and cut and price for the pickup and delivery problem with time windows". In: *Transportation Science* 43.3, pp. 267–286.
- Ropke, Stefan and David Pisinger (2006). "An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows". In: *Transportation Science* 40.4, pp. 455–472.
- Rudolph, Christian (2016). "How may incentives for electric cars affect purchase decisions?" In: *Transport Policy* 52, pp. 113–120.
- Ruiz, Efrain, Valeria Soto-Mendoza, Alvaro Ernesto Ruiz Barbosa, and Ricardo Reyes (2019). "Solving the open vehicle routing problem with capacity and distance constraints with a biased random key genetic algorithm". In: *Computers & Industrial Engineering* 133, pp. 207–219.
- Ruiz, Rubén and Thomas 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, Rubén and José Antonio Vázquez-Rodríguez (2010). "The hybrid flow shop scheduling problem". In: *European Journal of Operational Research* 205.1, pp. 1–18.
- Sadowskit, Wieslaw (1959). "a Few Remarks on the Assortment Problem". In: *Management Science* 6.1, pp. 13–24.
- Saeedvand, Saeed, Hadi S Aghdasi, and Jacky Baltés (2020). "Novel hybrid algorithm for Team Orienteering Problem with Time Windows for rescue applications". In: *Applied Soft Computing* 96, p. 106700.
- Saenz, Maria Jesus, Xenophon Koufteros, Jan Olhager, Sebastian Pashaei, and Henrik Sternberg (2015). "Design of global production and distribution networks". In: *International Journal of Physical Distribution & Logistics Management*.
- Sánchez-Oro, Jesús, Ana D López-Sánchez, and J Manuel Colmenar (2020). "A general variable neighborhood search for solving the multi-objective open vehicle routing problem". In: *Journal of Heuristics* 26.3, pp. 423–452.
- Santos, Douglas O and Eduardo C Xavier (2015). "Taxi and ride sharing: A dynamic dial-a-ride problem with money as an incentive". In: *Expert Systems with Applications* 42.19, pp. 6728–6737.
- Santos, Mário S., Thomás V.B. Pinto, Ênio Lopes Júnior, Luciano P. Cota, Marcone J.F. Souza, and Thiago A.M. Euzébio (2020). "Simheuristic-based decision support system for efficiency improvement of an iron ore crusher circuit". In: *Engineering Applications of Artificial Intelligence* 94, p. 103789.
- Sarmiento, Ana Maria and Rakesh Nagi (1999). "A review of integrated analysis of production-distribution systems". In: *IIE transactions* 31.11, pp. 1061–1074.
- Sauré, Denis and Zeevi Assaf (2013). "Optimal dynamic assortment planning with demand learning". In: *Manufacturing and Service Operations Management* 15.3, pp. 387–404.
- Savelsbergh, Martin WP and Marc Sol (1995). "The general pickup and delivery problem". In: *Transportation Science* 29.1, pp. 17–29.
- Schaller, Bruce (2017). "Unsustainable? The growth of app-based ride services and traffic, travel and the future of New York City". In: *Schaller Consulting*.
- Schnurr, Benedikt, Alexandra Brunner-Sperdin, and Nicola E Stokburger-Sauer (2017). "The effect of context attractiveness on product attractiveness and product quality: the moderating role of product familiarity". In: *Marketing Letters* 28.2, pp. 241–253.
- Schrage, Linus (1981). "Formulation and structure of more complex/realistic routing and scheduling problems". In: *Networks* 11.2, pp. 229–232.
- Schrank, David, Bill Eisele, and Tim Lomax (2019). *2019 Urban Mobility Report*. <https://static.tti.tamu.edu/tti.tamu.edu/documents/mobility-report-2019.pdf>, Last accessed on 2020-09-02.

- Schreieck, Maximilian, Hazem Safetli, Sajjad Ali Siddiqui, Christoph Pflügler, Manuel Wiesche, and Helmut Krcmar (2016). "A Matching Algorithm for Dynamic Ridesharing". In: *Transportation Research Procedia*. Vol. 19. Elsevier B.V., pp. 272–285.
- Scott, Judy and Carlton Scott (2017). "Drone delivery models for healthcare". In: *Proceedings of the 50th Hawaii international conference on system sciences*.
- Sharma, Satyendra Kumar, Srikanta Routroy, and Utkarsh Yadav (2018). "Vehicle routing problem: recent literature review of its variants". In: *International Journal of Operational Research* 33.1, pp. 1–31.
- Silva, Bhagya Nathali, Murad Khan, and Kijun Han (2018). "Towards sustainable smart cities: A review of trends, architectures, components, and open challenges in smart cities". In: *Sustainable Cities and Society* 38, pp. 697–713.
- Simonetto, Andrea, Julien Monteil, and Claudio Gambella (2019). "Real-time city-scale ridesharing via linear assignment problems". In: *Transportation Research Part C: Emerging Technologies*.
- Smith, Stephen A. and Narendra Agrawal (2000). "Management of Multi-Item Retail Inventory Systems with Demand Substitution". In: *Operations Research* 48.1, pp. 50–64.
- Snyder, Lawrence (2006). "Facility Location Under Uncertainty: A Review". In: *IIE Transactions* 38.7, pp. 547–564.
- Soler-Dominguez, Amparo, Angel A Juan, and Renatas Kizys (2017). "A survey on financial applications of metaheuristics". In: *ACM Computing Surveys (CSUR)* 50.1, pp. 1–23.
- Sombuntham, Pandhapon and Voratas Kachitvichayanukul (2010). "A particle swarm optimization algorithm for multi-depot vehicle routing problem with pickup and delivery requests". In: *World Congress on Engineering 2012. July 4-6, 2012. London, UK*. Vol. 2182. Citeseer, pp. 1998–2003.
- Song, Mei-xian, Jun-qing Li, Yun-qi Han, Yu-yan Han, Li-li Liu, and Qun Sun (2020). "Metaheuristics for solving the vehicle routing problem with the time windows and energy consumption in cold chain logistics". In: *Applied Soft Computing* 95, p. 106561.
- Sörensen, Kenneth and Fred Glover (2013). "Metaheuristics". In: *Encyclopedia of operations research and management science* 62, pp. 960–970.
- Soysal, Mehmet (2016). "Closed-loop Inventory Routing Problem for returnable transport items". In: *Transportation Research Part D: Transport and Environment* 48, pp. 31–45.
- Stecke, Kathryn E and Xuying Zhao (2007). "Production and transportation integration for a make-to-order manufacturing company with a commit-to-delivery business mode". In: *Manufacturing & Service Operations Management* 9.2, pp. 206–224.
- Stiglic, Mitja, Niels Agatz, Martin Savelsbergh, and Mirko Gradisar (2015). "The benefits of meeting points in ride-sharing systems". In: *Transportation Research Part B: Methodological* 82, pp. 36–53.
- (2018). "Enhancing urban mobility: Integrating ride-sharing and public transit". In: *Computers & Operations Research* 90, pp. 12–21.
- Su, Sheng, Fangzheng Zhou, and Haijie Yu (2019). "An artificial bee colony algorithm with variable neighborhood search and tabu list for long-term carpooling problem with time window". In: *Applied Soft Computing*, p. 105814.
- Sun, Minghe (2006). "Solving the uncapacitated facility location problem using tabu search". In: *Computers & Operations Research* 33.9, pp. 2563–2589.
- Svangren, Michael K, Mikael B Skov, and Jesper Kjeldskov (2018). "Passenger trip planning using ride-sharing services". In: *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, pp. 1–12.
- Talbi, El-Ghazali (2009). *Metaheuristics: from design to implementation*. Vol. 74. John Wiley & Sons.
- Tang, Lei, Zongtao Duan, and Yaling Zhao (2019). "Toward using social media to support ridesharing services: challenges and opportunities". In: *Transportation Planning and Technology* 42.4, pp. 355–379.

- Tasan, A. Serdar and Mitsuo Gen (2012). "A genetic algorithm based approach to vehicle routing problem with simultaneous pick-up and deliveries". In: *Computers & Industrial Engineering* 62, pp. 755–761.
- Tavakkoli-Moghaddam, Reza, Mohammadreza Meskini, Hadi Nasser, and Haed Tavakkoli-Moghaddam (2019). "A Multi-Depot Close and Open Vehicle Routing Problem with Heterogeneous Vehicles". In: *2019 International Conference on Industrial Engineering and Systems Management (IESM)*. IEEE, pp. 1–6.
- Tcha, Dong-wan and Bum-il Lee (1984). "A branch-and-bound algorithm for the multi-level uncapacitated facility location problem". In: *European Journal of Operational Research* 18.1, pp. 35–43.
- Technical Specification Group Services and System Aspects (2019). *Enhancement of 3GPP support for V2X scenarios; Stage 1 (Release 16)*. Technical Specification 3GPP TS 22.186 v16.2.0. 3rd Generation Partnership Project.
- (2020). *Service requirements for V2X services; Stage 1 (Release 16)*. Technical Specification 3GPP TS 22.185 v16.0.0. 3rd Generation Partnership Project.
- Thomas, Douglas J and Paul M Griffin (1996). "Coordinated supply chain management". In: *European Journal of Operational Research* 94.1, pp. 1–15.
- Thornhill, Adrian, Mark Saunders, and Philip Lewis (2009). *Research Methods for Business Students*. Prentice Hall: London.
- Ting, Chuan-Kang and Xin-Lan Liao (2013). "The selective pickup and delivery problem: formulation and a memetic algorithm". In: *International Journal of Production Economics* 141.1, pp. 199–211.
- Ting, Chuan-Kang, Xin-Lan Liao, Yu-Hsuan Huang, and Rung-Tzuo Liaw (2017). "Multi-vehicle selective pickup and delivery using metaheuristic algorithms". In: *Information Sciences* 406, pp. 146–169.
- Tordecilla-Madera, Rafael, Andrés Polo Roa, John Willmer Escobar, and Nicolas Clavijo Buritica (2018). "A mathematical model for collecting and distributing perishable products by considering costs minimisation and CO₂ emissions". In: *International Journal of Services and Operations Management* 31.2, pp. 207–234.
- Toth, Paolo and Daniele Vigo (1997). "An exact algorithm for the vehicle routing problem with backhauls". In: *Transportation science* 31.4, pp. 372–385.
- (2014). *Vehicle routing: problems, methods, and applications*. SIAM.
- Tranfield, David, David Denyer, and Palminder Smart (2003). "Towards a methodology for developing evidence-informed management knowledge by means of systematic review". In: *British Journal of Management* 14.3, pp. 207–222.
- Ulu, Canan, Dorothée Honhon, and Aydın Alptekinoğlu (2012). "Learning consumer tastes through dynamic assortments". In: *Operations Research* 60.4, pp. 833–849.
- United States Environmental Protection Agency (2020). *Fast Facts on Transportation Greenhouse Gas Emissions*. <https://nepis.epa.gov/Exe/ZyPDF.cgi?Dockey=P100ZK4P.pdf>, Last accessed on 2020-09-04.
- Ushizima, Daniela M, Fátima NS de Medeiros, Paulo HC Oliveira, and Andrea GC Bianchi (2020). "An Iterated Local Search-Based Algorithm to Support Cell Nuclei Detection in Pap Smears Test". In: *Enterprise Information Systems: 21st International Conference, ICEIS 2019, Heraklion, Crete, Greece, May 3–5, 2019, Revised Selected Papers*. Vol. 378. Springer Nature, p. 78.
- Verhoef, Peter C., P. K. Kannan, and J. Jeffrey Inman (2015). "From Multi-Channel Retailing to Omni-Channel Retailing. Introduction to the Special Issue on Multi-Channel Retailing." In: *Journal of Retailing* 91.2, pp. 174–181.
- Verhoef, Peter C., Andrew T. Stephen, P. K. Kannan, Xueming Luo, Vibhanshu Abhishek, Michelle Andrews, Yakov Bart, Hannes Datta, Nathan Fong, Donna L. Hoffman, Mandy Mantian Hu, Tom Novak, William Rand, and Yuchi Zhang (2017). "Consumer Connectivity in a Complex,

- Technology-enabled, and Mobile-oriented World with Smart Products". In: *Journal of Interactive Marketing* 40, pp. 1–8.
- Vidal, Carlos J and Marc Goetschalckx (1997). "Strategic production-distribution models: A critical review with emphasis on global supply chain models". In: *European Journal of Operational Research* 98.1, pp. 1–18.
- Vidal, Thibaut, Gilbert Laporte, and Piotr Matl (2019). "A concise guide to existing and emerging vehicle routing problem variants". In: *European Journal of Operational Research*.
- Villarinho, Pedro A, Javier Panadero, Luciana S Pessoa, Angel A Juan, and Fernando L Cyrino Oliveira (2021). "A simheuristic algorithm for the stochastic permutation flow-shop problem with delivery dates and cumulative payoffs". In: *International Transactions in Operational Research* 28.2, pp. 716–737.
- Wan, Wan Mazlina, Muhamad Syaiful Azrie, and Adibah Shuib (2018). "Mechanism For Drones Delivering The Medical First Aid Kits". In: *Advances in Transportation and Logistics Research* 1.1, pp. 303–315.
- Wang, De-Yun, Olivier Grunder, and Abdellah El Moudni (2015). "Integrated scheduling of production and distribution operations: a review". In: *International Journal of Industrial and Systems Engineering* 19.1, pp. 94–122.
- Wang, Jing Peng, Xuegang (Jeff) Ban, and Hai Jun Huang (2019). "Dynamic ridesharing with variable-ratio charging-compensation scheme for morning commute". In: *Transportation Research Part B: Methodological*.
- Wang, K, WQ Ma, H Luo, and H Qin (2016). "Coordinated scheduling of production and transportation in a two-stage assembly flowshop". In: *International Journal of Production Research* 54.22, pp. 6891–6911.
- Wang, Xing, Niels Agatz, and Alan Erera (2017a). "Stable matching for dynamic ride-sharing systems". In: *Transportation Science* 52.4, pp. 850–867.
- Wang, Xingyin, Stefan Poikonen, and Bruce Golden (2017b). "The vehicle routing problem with drones: several worst-case results". In: *Optimization Letters* 11.4, pp. 679–697.
- Wang, Yazhe, Baihua Zheng, and Ee-Peng Lim (2018). "Understanding the effects of taxi ride-sharing—A case study of Singapore". In: *Computers, Environment and Urban Systems* 69, pp. 124–132.
- Wassan, N (2007). "Reactive tabu adaptive memory programming search for the vehicle routing problem with backhauls". In: *Journal of the Operational Research Society* 58.12, pp. 1630–1641.
- Waters, C Donald J (2003). *Logistics: an introduction to supply chain management*. Palgrave Macmillan.
- Wei, Q., L. Wang, and A. Fei (2019). "Energy Minimization for Infrastructure-to-Vehicle Communications with Multiple Roadside Units". In: *2019 11th International Conference on Wireless Communications and Signal Processing (WCSP)*, pp. 1–5.
- Wu, Weiqin, Yu Tian, and Tongdan Jin (2016). "A label based ant colony algorithm for heterogeneous vehicle routing with mixed backhaul". In: *Applied Soft Computing* 47, pp. 224–234.
- Xu, Huayu, Fernando Ordóñez, and Maged Dessouky (2015). "A traffic assignment model for a ridesharing transportation market". In: *Journal of Advanced Transportation* 49.7, pp. 793–816.
- Yan, Shangyao and Chun-Ying Chen (2011). "A model and a solution algorithm for the car pooling problem with pre-matching information". In: *Computers & Industrial Engineering* 61.3, pp. 512–524.
- Yang, Fangchun, Shangguang Wang, Jinglin Li, Zhihan Liu, and Qibo Sun (2014). "An overview of internet of vehicles". In: *China communications* 11.10, pp. 1–15.
- Yang, Ming-Hsien (2001). "An efficient algorithm to allocate shelf space". In: *European Journal of Operational Research* 131.1, pp. 107–118.
- Yang, Ming-Hsien and Wen-Cher Chen (1999). "A study on shelf space allocation and management". In: *International journal of production economics* 60, pp. 309–317.

- Yousefikhoshbakht, Majid and Azam Dolatnejad (2017). "A column generation for the heterogeneous fixed fleet open vehicle routing problem". In: *International Journal of Production Management and Engineering* 5.2, pp. 55–71.
- Yu, Guanding, Qimei Chen, and Rui Yin (2014). "Dual-threshold sleep mode control scheme for small cells". In: *IET Communications* 8.11, pp. 2008–2016.
- Yu, Vincent F, Sesya Sri Purwanti, AAN Perwira Redi, Chung-Cheng Lu, Suprayogi Suprayogi, and Parida Jewpanya (2018). "Simulated annealing heuristic for the general share-a-ride problem". In: *Engineering Optimization* 50.7, pp. 1178–1197.
- Yu, Yang, Yuting Wu, and Junwei Wang (2019). "Bi-objective green ride-sharing problem: Model and exact method". In: *International Journal of Production Economics* 208, pp. 472–482.
- Zachariadis, Emmanouil E and Chris T Kiranoudis (2012). "An effective local search approach for the vehicle routing problem with backhauls". In: *Expert Systems with Applications* 39.3, pp. 3174–3184.
- Zeng, Zheng-yang, Wei-sheng Xu, Zhi-yu Xu, and Wei-hui Shao (2014). "A hybrid GRASP+ VND heuristic for the two-echelon vehicle routing problem arising in city logistics." In: *Mathematical Problems in Engineering*.
- Zhang, Min, Chengshang Ren, G Alan Wang, and Zhen He (2018). "The impact of channel integration on consumer responses in omni-channel retailing: the mediating effect of consumer empowerment". In: *Electronic commerce research and applications* 28, pp. 181–193.
- Zhang, Xijun, Qirui Zhang, Zhanting Yuan, Chenhui Wang, and Lijuan Zhang (2020). "The Research on Planning of Taxi Sharing Route and Sharing Expenses". In: *Mathematical Problems in Engineering* 2020.
- Zhao, Wayne Xin, Sui Li, Yulan He, Edward Y. Chang, Ji Rong Wen, and Xiaoming Li (2016). "Connecting Social Media to E-Commerce: Cold-Start Product Recommendation Using Microblogging Information". In: *IEEE Transactions on Knowledge and Data Engineering* 28.5, pp. 1147–1159.
- Zhao, Xin Wayne, Yanwei Guo, Yulan He, Han Jiang, Yuexin Wu, and Xiaoming Li (2014). "We know what you want to buy: a demographic-based system for product recommendation on microblogs". In: pp. 1935–1944.
- Zhou, H., W. Xu, J. Chen, and W. Wang (2020). "Evolutionary V2X Technologies Toward the Internet of Vehicles: Challenges and Opportunities". In: *Proceedings of the IEEE* 108.2, pp. 308–323.