UNIVERSITAT POLITÈCNICA
DE CATALUNYA
BARCELONATECH

Optimized content caching strategies for multi-access edge computing (MEC)-assisted future cellular networks

Tadege Mihretu Ayenew

DOCTORAL THESIS

Optimized Content Caching Strategies for Multi-access Edge Computing (MEC)-assisted Future Cellular Networks

Author:
Tadege Mihretu Ayenew

Supervisors:
Prof. Lazaros Merakos
Dr. Nikos Passas
Prof. Luis G. Alonso Zaŕate

A dissertation submitted in fulfillment of the requirements for the degree of Doctor of Philosophy

in the

Wireless Communications and Technologies Research Group
Department of Signal Theory and Communications

May 13, 2023

Declaration of Authorship

I, Tadege Mihretu Ayenew, declare that this dissertation entitled: "Optimized Content Caching Policies for Multi-access Edge Computing (MEC)-assisted Future Cellular Networks", and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this university.
- Where any part of this dissertation has previously been submitted for a degree or any other qualification at this university or any other institution, this has been clearly stated.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this dissertation is entirely my work.
- I have acknowledged all main sources of help.
- Where the dissertation is based on work done by myself jointly with others, I have exactly clarified what was done by others and what I have contributed.

Signed: _____

Date: _____

Authorization by Supervisor

We, the undersigned, certify that we have read and recommend to the Doctoral School of Universitat Politècnica De Catalunya (BarcelonaTech) for acceptance of the dissertation entitled, "Optimized Content Caching Strategies for Multi-access Edge Computing (MEC)-assisted Future Cellular Networks," by Tadege Mihretu Ayenew:

Lazaros Merakos (Supervisor)
Informatics and telecommunication, National and Kapodistrian University of Athens, Greece

_____, Date:_____

Nikos Passas (Co-supervisor)
Informatics and telecommunication, National and Kapodistrian University of Athens, Greece

_____, Date:_____

Luis Gonzaga Alonso Zaŕate (Tutor)
Signal Theory and Communications, Universitat Politècnica De Catalunya, Spain

_____, Date:_____

Abstract

Handling the tsunami of multimedia content is a big challenge for heterogeneous cellular networks. Serving large volumes of content from the central system to end-users, through a bandwidth-limited network, at peak time leads to network congestion. Typically, popular contents impede network performance and incur a high cost because they are redundantly transmitted to users. Also, some contents are too far from users, causing service delays. Owing to the above reasons, the network performance deteriorates, and meeting the required quality of experience becomes difficult. One way to improve HCN performance is through the use of caching of popular contents. This research proposes and analyzes content caching optimization strategies for several scenarios of cellular networks enhanced with multi-access edge computing.

Several studies have shown that backhaul congestion and service latency can be reduced by applying efficient content caching policies. However, a challenging issue is that the network edges have limited resources, such as cache size and computation. Due to such limitations, how to effectively exploit the available network resource and cache contents optimally remains an open-ended question. Content caching strategies have been the subject of several research efforts; however, in most works, the accurate modeling of the caching problem is a challenging open problem. In addition, most of them employ intractable optimization tools, which depend on greedy algorithms and heuristics. Most strategies must also be scaled into large networks with computationally feasible solutions. More specifically, the complex joint-caching problem remains open.

This dissertation aims to fill these gaps and contribute to modeling real-network characteristics using advanced combinatorial optimization tools. We propose different content caching formulations and design novel optimization strategies. The general approach is to effectively model caching schemes using popularity-based selection problems and solve the models using efficient algorithms. We devise an optimal caching scheme using dynamic programming. A more complex but exact caching strategy is proposed, where heterogeneous caching edges are clustered to offload the central system. Furthermore, we propose three demand-aware caching strategies, where a source caches contents to multiple caching edges, and the content popularity differs for each edge. Finally, we propose a demand-aware joint caching problem modeling where the content placement and delivery are mutually optimized.

The performances of the proposed strategies are thoroughly evaluated using extensive system-level simulations, comparing them with existing content caching strategies. The numerical results show that the proposed strategies outperform baseline strategies in terms of relevant key performance indicators. Also, the associated MEC functionalities are computationally feasible. Finally, useful cache design guidelines are set for various wireless network scenarios, contributing to the future design of emerging technologies and practical business models.

Resumen

Gestionar el tsunami de contenidos multimedia existente en la actualidad es un gran reto para las redes celulares heterogéneas. Dar servicio a grandes volúmenes de contenidos desde un sistema central para los usuarios finales, a través de una red de ancho de banda limitado y en hora punta, puede provocar la congestión de la red. Normalmente, los contenidos más populares complican el rendimiento de la red y suponen un alto coste dado que se transmiten redundantemente a los diferentes usuarios. Además, algunos contenidos están físicamente lejos de los usuarios, lo que provoca retrasos en el servicio. Por todo ello, el rendimiento de la red se deteriora y resulta difícil ofrecer la calidad de experiencia requerida. Una forma de mejorar el rendimiento de las llamadas HCN es mediante el uso de la caché de contenidos populares. Esta investigación propone y analiza nuevas estrategias de optimización del almacenamiento en caché de contenidos para varios escenarios de redes celulares mejoradas incluyendo Edge Computing multiacceso.

Varios estudios han demostrado que la congestión del backhaul y la latencia del servicio pueden reducirse aplicando políticas eficientes de almacenamiento de contenidos en caché. Sin embargo, un problema es que los bordes de la red tienen recursos limitados, como el tamaño de la caché y la capacidad de cómputo. Debido a estas limitaciones, sigue siendo una cuestión pendiente estudiar cómo explotar eficazmente los recursos de red disponibles y almacenar en caché los contenidos de forma óptima. Las estrategias de almacenamiento en caché de contenidos ya han sido objeto de varios trabajos de investigación; sin embargo, en la mayoría de ellos, el modelado preciso del problema de almacenamiento en caché es un problema abierto y desafiante. Además, la mayoría de ellos emplean herramientas de optimización intratables en la práctica, que dependen de algoritmos heurísticos que requieren recursos de computación inasumibles. Además, la mayoría de las estrategias deben poder escalarse a redes de gran tamaño y tener soluciones factibles desde el punto de vista computacional. Más concretamente, el complejo problema del caché conjunto sigue sin resolverse.

Esta tesis pretende cubrir parcialmente estas lagunas y contribuir a modelar las características de las redes reales mediante herramientas avanzadas de optimización combinatoria. Proponemos diferentes formulaciones de almacenamiento en caché de contenidos y diseñamos novedosas estrategias de optimización. El enfoque general consiste en modelar eficazmente los esquemas de almacenamiento en caché mediante problemas de selección basados en la popularidad y resolver los modelos mediante algoritmos eficientes. Ideamos un esquema óptimo de almacenamiento en caché mediante programación dinámica. Se propone una estrategia de caché más compleja pero exacta, en la que se agrupan datos de caché heterogéneos almacenados en los bordes de la red para descargar el sistema central. Además, proponemos tres estrategias de almacenamiento en caché en función de la demanda, en las que una fuente almacena contenidos en caché en múltiples bordes de almacenamiento en caché y la popularidad del contenido difiere para cada borde. Por último, proponemos una modelización de problemas de caché conjunta en la que la ubicación y la entrega de contenidos se optimizan mutuamente.

Las prestaciones de las estrategias propuestas se evalúan exhaustivamente mediante extensas simulaciones a nivel de sistema, comparándolas con las estrategias de almacenamiento en caché de contenidos existentes. Los resultados numéricos muestran que las estrategias propuestas superan a las estrategias de referencia en términos de indicadores clave de rendimiento. Además, las funcionalidades MEC asociadas son factibles desde el punto de vista computacional. Por último, se establecen directrices útiles de diseño de cachés para diversos escenarios de redes inalámbricas, lo que contribuye al futuro diseño de tecnologías emergentes y modelos de negocio prácticos.

Acknowledgements

Foremost, I thank The Almighty God for enabling me to accomplish this Ph.D. under a series of tough challenges.

I want to thank my supervisor Prof. Lazaros Merakos and co-supervisor Dr. Nikos Passas, for all their help and advice during my entire research stay at the National and Kapodistrian University of Athens (NKUA), Greece. I also thank Prof. Luis Gonzaga Alonso Zaŕate, my tutor at Universitat Politècnica De Catalunya (UPC), Spain, for his relentless guidance and motivating feedback. Their supervision, support, and teaching empowered me to achieve this dissertation.

My gratitude goes to the European Commission for its funding, training me as an Early Stage Researcher, under the auspice of Marie Skłodowska Curie Innovative Training Network (MSCA ITN-H2020) program of the SPOTLIGHT project, at the Informatics and Telecommunication Department of NKUA. I want to thank Dr. Dionysis Xenakis, my daily supervisor, for his technical support in achieving my research works. I thank all staff members of the GAIN Group in the department, mainly Subin N., for your great companionship. The incredible managerial support from Judith López, at the Doctoral School of UPC, enabled me to stay on track with the long process.

I want to thank my Ethiopian friends in Athens, who guarded me during my crisis times. I sincerely thank those who opened their doors when I was homeless for several months; impossible to mention all but Fikir & Seni, Eyobe & Sebli, Zinu & Belu, Fanos, and Alem made it special! A typical thank goes to my priest Fr. Gebreiyesus Abebe (Komos), for his continuous mentoring. I want to thank all my fellow Mahibere Kidusan-Europe members for invigorating and always being beside me.

I would also thank my close and casual friends for your kind words, support, and motivation. You were there for me when I was desperate and distressed, especially during the COVID-19 pandemic and the traumatic Ethiopian conflicts. Among several, allow me to thank: Dr. Arega G., Dr. Sosina M., Dr. Alemnew S., Dr. Abreham Z., and Dr. Menor T. for encouraging and giving me valuable feedback in achieving this thesis. Special thank goes to my mentors, Dr. Belachew C. and Dr. Fr. Yabibal M.

Last but not least, I would like to thank my two extended families for their love and inspiration. Aunt Ethiopia D., representing two sides, please accept my appreciation. My darling, Firehiwot B., thank you for energizing me lastly.

I praise all the saints, mainly St. John the Theologian and St. Mary Theotokos, for Their intercession and quelling the odd moments!

Contents

List of Figures

List of Tables

List of Abbreviations

ASE	**A**rea **S**pectral **E**fficiency
B5G	**B**eyond **5G** wireless
BB-ZoMKP	**B**ound-and-bound **Z**ero-one **M**ultiple **K**napsack **P**roblems
BOP	**B**ilevel **O**ptimization **P**roblem
bps	**b**its **p**er **s**econd
CDN	**C**ontent **D**elivery **N**etwork
CHP	**C**ache **H**it **P**robability
CHR	**C**ache **H**it **R**atio
CS	**C**entral **S**erver
DASH	**D**ynamic **A**daptive **S**treaming over **H**TTP
DJ-BOP	**D**emand-aware **J**oint caching using **BOP**
DP	**D**ynamic **P**rograming
DP-ZoSKP	**DP**-based **Z**ero-one **S**ingle **K**napsack **P**roblems
D-MKP	**D**emand-aware **M**ultiple **K**napsack **P**roblem
D-ZoSAP	**D**emnad-aware**Z**ero-one **S**eparable **A**ssignement **P**roblems
D2D	**D**evice-to-**D**evice
EE	**E**nergy **E**fficiency
EB	**E**xa**b**yte
ETSI	**E**uropean **T**elecommunications **S**tandards **I**nstitute
FIFO	**F**irst **I**n **F**irst **O**out
FWA	**F**ixed **W**ireless **A**access
HCN	**H**eterogeneous **C**ellular **N**etwork
HTTPS	**H**ypertext **T**ransfer **P**rotocol **S**ecure
ICN	**I**nformation **C**entric **N**etworks
IoT	**I**nternet of **T**hings
ITU	**I**nternational **T**ellecommunication **U**union
LFU	**L**east **F**requently **U**sed
LRU	**L**east **R**ecently **U**sed
LTE	**L**ong **T**erm **E**volution
MBS	**M**ain **B**ase **S**tation
MEC	**M**ulti-access **E**dge **C**omputing
MH	**M**obile **H**elpers
MINLP	**M**ixed **I**nteger **N**onlinear **P**rogramming
MKP	**M**ultiple **K**napsack **P**roblem
ML	**M**achine **L**earning
mMTC	**M**assive **M**achine **T**ype **C**communication
MNO	**M**obile **N**etwork **O**perators

MPTCP	**M**ulti **P**ath **T**ransport **T**ransport **P**rotocol
MP-QUIC	**M**ulti **P**ath **Q**uic **UDP** **I**internet **C**onnections
MTC	**M**achine **T**ype **C**communication
NR	**N**ew **R**adio
QoE	**Q**uality of **E**xperience
QoS	**Q**uality of **S**ervice
QUIC	**Q**uick UDP **I**nternet **C**onnections
RAN	**R**adio **A**ccess **N**etwork
SAP	**S**eparable **A**ssignement **P**roblem
SBS	**S**ub **B**ase **S**tation
SMKP	**S**pecial **M**ultiple **K**napsack **P**roblem
SIR	**S**ignal-to- **I**terefernce **R**atio
TCP	**T**ransport **C**ontrol **P**rotocol
UDP	**U**ser **D**atagram **P**rotocol
UE	**U**ser **E**quipment
UMTS	**U**niversal **M**obile **T**elecommunications **S**ystem
VoD	**V**ideo **o**n **D**emand
V2X	**V**ehicular-to-everything
ZoMKP	**Z**ero-one **M**ultiple **K**napsack **P**roblems
ZoSAP	**Z**ero-one **S**eparable **A**ssignment **P**roblems
ZoSKP	**Z**ero-one **S**ingle **K**napsack **P**roblems
3GPP	**3**rd **G**eneration **P**artnership **P**roject
3G	3rd **G**eneration of cellular network technology
4G	4th **G**eneration of broadband cellular network technology
5G	5th **G**eneration of broadband cellular network technology

List of Symbols

f_m	Content unique identifier
n	Mobile helper unique identifier
$\mathcal{N}$	Set of mobile helpers: $\mathcal{N} = \{n : n = 1, ..., \lvert\mathcal{N}\rvert\}$
$\rho_{m,n}$	Popularity of content f_m at $n^{th} MH$
$\mathcal{P}$	Popularity matrix of all f_m at all MHs: $\mathcal{P}$=$\{\rho_{n,m} : \forall f_m \in \mathcal{M}, \forall n \in \mathcal{N}\}$
s_m	Size of content f_m
$\mathcal{M}$	Content library at MBS: $\mathcal{M} = \{f_m : m = 1, ..., \lvert\mathcal{M}\rvert\}$
ξ	Cache mean size of mobile helpers
$\phi_{m,n}$	Request probability of content f_m by users associated to helper n
Ψ	Cache hit probability (CHP) of the cluster
Φ	Cache hit ratio (CHR) of the cluster
Ω	Cache service reward
μ	Content size distribution mean value
$\mathcal{S}$	Content size vector: $\mathcal{S} = \{s_m : m = 1, ..., \lvert\mathcal{M}\rvert\}$
$r_{m,n}$	Request rate to content f_m through n^{th} MH
$\mathcal{R}$	All request rate through all MHs f_m: $\mathcal{R}$=$\{r_{n,m} : \forall f_m \in \mathcal{M}, \forall n \in \mathcal{N}\}$
u	User unique identifier
δ	User distribution mean value
$\mathcal{U}$	Set of users: $\mathcal{U} = \{1, ...u, ..., \lvert\mathcal{U}\rvert\}$
σ^2	Cache size distribution variance
γ	Content popularity distribution Zipf parameter
$x_{m,n}$	Caching decision indicator of f_m to n^{th} MH
$\mathbf{x}$	Set of indicators: $\mathbf{x} = \{x_{m,n}, \forall f_m \in \mathcal{M}, n \in \mathcal{N}\}$
$v_{m,u}$	Service utility cost while serving f_m to user u
$\Upsilon(\mathbf{x})$	Total utility service cost for all contents decided by $\mathbf{x}$

Dedicated to my parents:

Mihretu Ayenew & Workie Bogale
and
Yigrem Shitaye & Fentanesh Alamineh.

Chapter 1

Introduction

The current heterogeneous cellular networks (HCN) is becoming complex in kind and the service types it delivers. It incorporates diversified devices with different computational capabilities and service quality requirements. In addition, the demand for traffic-intensive applications is increasing on the user side, which operates resource-consuming smart devices. Though HCN will be a major component of the Internet, due to its flexibility and low cost of deployment, some of its outstanding performance bottlenecks remain unresolved [1].

The monthly traffic volume of multimedia contents (e.g., video, photos, and audio) in mobile networks, including fixed wireless access (FWA), is steadily plunging at an exponential rate. The mobility report by Ericsson showed that the monthly global mobile data traffic exponentially increased from 13.5 exabytes in 2018 to 100 Exabytes (EB) in 2022. This figure is double mobile traffic in 2021 and is predicted to reach 300 EB by 2026. The current annual mobile data traffic growth is 40% [2]. In addition, due to dynamic situations such as the COVID-19 pandemic, services such as video conferencing, streaming, office software suites, and web browsing have become unprecedentedly common and have accelerated the extensive use of online video streaming services and teleworking.

This mobile data traffic proliferation is triggered by an explosive surge of smart devices in the market and fast subscriptions to services that are running on cellular devices. It is further enhanced by the emerging Internet of Things (IoT) development. The wide penetration of smart devices, with various computational and storage capacities, has led to the exponential release of multimedia content [3]. In parallel, we are witnessing an overwhelming growth of traffic-demanding applications that deliver complex multimedia services such as video on demand (VoD), augmented reality, immersive media formats, and online gaming. For example, the current average monthly data usage is 10 GB per user and is predicted to be 35 GB by 2026 [4], where the fifth generation (5G) network will cover nearly 53% of total traffic.

The future generation cellular networks, hit hard by the tsunami of Internet content, must have orchestrated communication lines to avoid performance marring issues. To that extent of orchestration, several standardizations are set as a reference to various performance indicators, which are becoming hard to meet practically. Among many efforts, an application-layer *traffic management* has been the central focus of the research in wireless communication for the past decade. It ranges from understanding the contextual growth of data traffic to alleviating network congestion.

From an application-layer perspective, we need to improve the performance of cellular networks' traffic management by applying advanced content processing technologies, but not limited to, that help meet the required quality of experience (QoE) [5]. Among some potential technologies, *content caching* is proposed as a radical approach to alleviating the foreseen content traffic problems [6]. It has substantially boosted the network performance metrics such as throughput, spectrum usage, per-link spectral efficiency, and spatial reuse. In short, caching refers to prefetching contents from the central head of mobile network operators (MNO) library to the timely storing caching edges, close to end users. The source of contents might be the main base station (MBS), while the caching edge can be any cache-capable network device called a mobile helper (MH). Though a well-studied topic in the case of content delivery networks (CDN) and the Internet, cellular content caching is an open-ended research topic; in fact, one of the disruptive technologies to actualize the 5G and beyond cellular networks.

Actually, not alone the content caching brings the required QoE, but there are other key enabling technologies and network-edge capabilities. For that matter, the 5G network edges are becoming capable of caching large components of the traffic and executing content processing and computing tasks on the radio metadata. In the near future, the third-generation chipsets will enable performance-optimized 5G devices to reach the low-cost network segments. This refers to multi-access edge computing (MEC) technology that exploits the devices' resources, such as storage and computation, which significantly enhances the content caching in the HCN [7].

The MEC was introduced by European Telecommunications Standards Institute (ETSI) for leveraging the 5G network where the edge devices provide IT service environments and computing capabilities [8]. This technology brings the computational capacity of the central system (CS), of distant processors in the classical network, to the very close proximity of end users. It reduces the computation efforts for content caching and optimization so as to achieve ultra-low latency and high bandwidth requirements [9]. In addition, the real-time preprocessing of contents and reactive information processing services boost the practicability of caching for service providers and application developers who can embed it into the network.

Recently, distributed content clouds and caching devices have been adapted to fetch contents from the network CS to last-mile edges. Thankfully, these cloud architectures and caching devices are emerging with higher capabilities to ease the communication strains among devices without central auspices. While enjoying the merits of *distributed caching*, it is also worth setting up a cooperative system. This research aims at exploiting optimization techniques on cooperative content caching to reduce network resource consumption while improving QoE.

1.1 Motivation and Objectives

In this section, the problem statement and motivation of the research are explained, followed by the research objectives.

1.1.1 Problem statement and motivation

Multimedia content service through the cellular infrastructure has seen a paradigm shift due to the tsunami of mobile video traffic generated by billions of smart devices populating the network. As a result, Internet service providers (ISP) and content delivery networks (CDN) have seen the unprecedented challenges of maintaining the esteemed QoE. From a traffic content management point of view, it is important to mention a few challenges that need to be addressed in future cellular networks.

The first challenge arises from the explosive release of traffic data being held on a non-flexible and resource-constrained network. Mainly, the backhaul and fronthaul networks have limited link capacity, which is the main obstruction to getting the required user QoE and network quality of service (QoS). The huge data traffic that is simultaneously circulating in this network creates drastic congestion and download delay, which is an ever-prevailing challenge.

The second challenge is imposed by most frequently requested, identified as *popular*, contents that dominantly exploit the cellular network. A critical mass of mobile content requests refers to the level of consumption of the same popular content by different end devices spanning small geographical regions. Such a concentrated critical mass of requests puts a tremendous burden on the backhaul of content-agnostic cellular networks. Upon requests, these popular video contents are redundantly retransmitted from the backhaul server, alternatively called the central system, to the end user equipment (UE), using the bandwidth-limited backhaul at peak time. This incurs higher transmission costs to the constrained network.

The third challenge comes from the unbeatable distances between the CDN and user UEs across the corners of the globe. This is contrary to the need for extremely low latency service applications, still poising overwhelmingly increasing traffic demand. Unlikely, contents served from the backhaul CS, which are quite distant servers on the Internet, pass through multiple routers before they are delivered to users. This process forces the real network to exhibit a high response delay of up to seconds, which is far from the set goal of 1 ms or less latency for the 5G and beyond networks.

The last foreseen challenge is that intensive end-user applications are creating big-sized and dynamically changing contents that consume the network resources. Unlike the CDN protocol, these contents are produced in a dispensed manner, so they make the caching process more complex. One viable solution is applying appropriate content preprocessing on network edges before caching. However, the network and the edges have finite resources, such as storage memory and computing. Hence, the HCN needs processing techniques that effectively exploit these scarce resources to reduce the burden of huge contents while guaranteeing QoE.

In contrast to the above-mentioned challenges, in-network storage technologies are exponentially advancing while their price is decaying fast. Having this opportunity against the defined bottlenecks, content caching becomes a viable solution to offloading network congestion, reducing retransmission costs, and avoiding service latency. However, the caching strategy should take into consideration of the limited resources of the network. Therefore, we inevitably question: 'Which contents to cache in priority?', 'where to cache the content?', 'when to make the caching process?'.

Although vast research has been done in the state of the art, the caching models have gaps in effectively capturing the true nature of the cellular network. Because of the complexity of the caching problem, we notice that the existing problem representations focus on probabilistic and intractable models. In addition, most of them model the content caching process with a single decision-based strategy for the sake of end-to-end optimization, which becomes an unsolvable NP-hard problem. Even though some solid works propose two decision-based joint caching strategies, they apply a duality property of the content placement and delivery phases' nested relationship. However, the phases' relationship does not fulfill a dual-nested property. On the other hand, some existing strategies fail to treat heterogeneities in network attributes such as content popularity, content size, and cache sizes. On top of the modeling gaps, most strategies employ greedy and heuristic solutions, which are far from optimality. Other solutions have limitations of scalability for large real networks. To address these pressing issues of caching in the HCN, we conduct extensive research on content caching scenarios in increasing order of their complexity.

1.1.2 Research objectives

Having a list of motivating factors and the throttles of expected QoE, the thesis aims to propose practical and cooperative caching optimization strategies for cluster-centric 5G and beyond cellular networks. Specific objectives of the research are:

- analyze the state-of-art content caching and resource utilization problems, model strategies, and respective optimization solutions,
- design an optimal content caching strategy for the MEC-enabled 5G and beyond cellular networks, considering their future capabilities and limitations. Scale up the proposed strategy to other communication protocols and industrial use cases,
- design a novel resource utilization strategy for constrained MEC-enabled cellular networks, with different collaboration schemes, which is an extensional optimization of the caching for classic cellular networks,
- research on an alternative and future caching policy and exploit opportunities of convenient protocols for enhancing its performance.

1.2 Related Works on Caching to Improve QoE

Because content caching at cellular networks has been a point of attention for nearly a decade, we have sufficient literature on the strategies. Besides, content caching remains an active research topic due to its complexity and continuing proliferation of traffic. The ultimate goal of caching strategies is to increase the performance of the HCN, in terms of key performance indicators (KPI), and to satisfy users' aggressive demand for data traffic. For a better insight into the literature, we explain the cooperative caching strategy and build it up with joint and independent caching schemes. Side-by-side, we

revise the mathematical tools used to model caching problems. Following it, we review the optimization tools that are used to solve these models. In the end, we review the MEC-based caching strategies.

1.2.1 Cooperative and non-cooperative caching

There are two categories of caching processes in existing modeling and performance analysis for any caching system. The first category is *non-cooperative*, which acts that each MH autonomously selects its content to the cache using its algorithm. The second one is *cooperative*, which assumes centrally planned content placement and delivery across multiple caching edges. The authors in [10], [11] have shown that cooperative caching improves network performance due to the enhanced utilization of edge computing and storage resources. However, they have short of assuming content size diversity and random request distribution. Under the same assumptions, authors in [12] achieve a lower download latency and better storage utilization, where the femto-base stations (FBS) and UEs act as MH.

The work in [13] maximizes the area spectral efficiency by using an independent caching strategy to offload the backhaul capacity for dense networks, where missed contents are directly served from the CS. However, this approach was spearheaded by cooperative caching, which helps the network edges to leverage the distribution of contents within a specified cellular network area and brings a shared workload or congestion offloading [14]. In cooperative caching, the entire decision in the system is made by a central entity, called *central head*. Hence, caching mobile helpers cooperate to cache required files and pass them to the user [15]–[19].

Authors in [16] focus on a cooperative caching strategy to show that, by caching files at an off-peak time, it is possible to enhance the network performance through a broadcasting technique. Also, authors in [20], [21] use this caching strategy to exploit the scarce edge storage and resource allocation in device-to-device (D2D) communication. Mainly, the work in [22] analyses the impact of cooperative caching and devised cache refreshment in D2D communication with data-intensive applications. In [23], the authors propose a cooperative and hierarchical caching strategy for the cloud RAN (C-RAN) case, where contents are jointly cached at the central baseband unit (BBU) and at the remote radio head (RRHs).

Within the cooperative scheme, some techniques use further clustering of the network devices based on parameters such as the requested file or the cooperation distance [18], [19], [24]. In light of the forthcoming MEC-RAN integration, several clustering techniques exist in the literature. These techniques consider different objective functions such as minimizing end-to-end service delay [25], reducing traffic congestion within the MEC cluster [26], enhancing MEC service coverage [27], and offloading core network traffic to the edge MEC edges [28]. Other strategies streamline distributed caching where the required files are stored at the different tiers of the network to increase the areal spectral efficiency, as shown in [29] and [17]. In [29], the authors develop a contention-based multimedia content delivery protocol, which avoids possible collision among concurrent transmissions by different active transmitters. This

maximizes the successful content delivery probability to the user equipment and the coverage probability.

1.2.2 Joint and separate caching

From a design perspective, the optimization of content placement and delivery phases have distinct interdependence. However, solving a joint optimization problem with an end-to-end approach is quite complex. In general, two methods are applied: joint optimization where content placement and delivery are treated simultaneously, and separate optimization where the placement is managed and the delivery process is optimized with a different scheme.

The authors in [10] propose a joint caching and delivery policy by transforming the nested-dual problem to an equivalent mixed integer non-linear programming (MINLP) problem that can be solved using the advanced branch-and-bound method. In this case, each MH decides its own content to the cache. Also, [30] proposed a joint and cooperative caching scheme to maximize the number of requests to be served without accessing the central head.

Other works propose independent optimization such that they mainly emphasize content placement and apply delivery enhancement methods. The work of [31] prioritizes finding an optimal cooperative content placement strategy. In [24], authors focus on proposing a near-optimal solution, using multicast, and they achieved better performance at a manageable complexity in the general region under mild conditions.

1.2.3 Caching-problem modeling tools

Content caching is a constrained problem usually modeled using different mathematical tools, based on the network architecture and considered objective functions [32]. These tools can be grouped into four categories.

The first and dominant category of content cache modeling is by optimization problems, various in type. The work in [33], [34] uses the knapsack problem to formulate content caching. Mainly, authors in [35] map the video caching to a multiple Knapsack problem, where the UEs have access to associated MH. Other optimization problems include mixed integer programming [36], linear integer programming [12], and mixed integer non-linear programming [10]. In [37], the authors model the content placement problem using the index coding algorithm, which assumes uncoded placement and coded delivery and derives an optimal lower-bound transmission rate. Authors in [38] use the Lyapunov model to optimize the average data rate with limited delay.

The second category of caching models uses probabilistic approaches, mainly stochastic geometry. The work in [11], [14], [24] uses stochastic modeling to increase the performance of HCN, in terms of a cache hit probability, area spectral efficiency, and service delay. The work in [39] applies stochastic geometry expression to improve the coverage probability. The third modeling category is using machine learning, which helps the CS make decisions based on information processing as analyzed in [40]. They apply machine learning tools to learn the content popularity and make proactive caching,

Proposed strategies			[49]. Song et al.	[50]. Jia et al.	[13]. Altieri et al.	[29]. Cui et al.	[16]. Fan et al.	[14]. Yang et al.	[51]. Wen et al.	[17]. Chen et al.	[52] Afshang et al.	[6]. Golrezaei et al.	[53]. Carlsson et al.
Network Caching Process	Content Placement	Cooperative caching		✓	✓			✓	✓		✓	✓	
		Hybrid caching				✓							
		Clustered caching			✓	✓	✓	✓		✓	✓	✓	
		Distributed caching	✓		✓		✓					✓	
		Centrally controlled system	✓							✓	✓	✓	
	Content Delivery	Coded information caching							✓			✓	
		Multicast or broadcast			✓	✓	✓	✓	✓				
		Coordinated multipoint					✓		✓		✓	✓	
		Transparent caching		✓									
Constraints & Interference handling		Successive cancelling									✓		
		Attenuation handling			✓								
		SINR threshold		✓			✓					✓	
Prediction based caching		Popularity prediction		✓									✓
		Position prediction										✓	
		Request prediction											✓

FIGURE 1.1: Summary table of proposed caching in the literature.

which resulted in a very high cache hit rate and backhaul offloading. Authors in [41], use transfer learning to train the content popularity and apply predictive and cooperative caching. Similarly, the Markov process is used in [42] to capture the dynamics of the network usage by the mining user's profile, such as location and request pattern, and predict the future behavior of network usage. Recent work in [43] uses deep learning to retrieve the content demand and proactively cache in a MEC-assisted network, to improve download delay for mission-critical services. However, from a business standpoint, it is yet under anticipation that pushing the MEC processing to the edge creates an over cost to the subscriber and affects the market.

The last cache modeling category uses game theory, which takes UEs as the player parties and objective functions as gains. Authors in [44] demonstrate that cost-effective and proactive caching schemes can be implemented using game theory, which reduces

the impact of UE selfishness. Authors in [45] exploit the Stackelberg game to model the interaction between caching edges and restrain their selfishness.

1.2.4 Constrained resource optimization solutions

Depending on the placement model type, different solving algorithms are also proposed. Theoretically, some of these problems are solved using closed-form techniques. Needless to say, most solutions to cache optimization problems rely on greedy and heuristic solutions. The work in [36] applies the Lagrangian relaxation and solves it using a heuristic approach, which gives a suboptimal solution.

On a popularity-based caching strategy in [14], they optimize caching using greedy solutions while [46] uses a greedy heuristic. Though the greedy and heuristic algorithms are simpler, they are sub-optimal in a practical sense [33]. In addition, they can not deal with the spatial difference of content popularity, capacity limitedness, and heterogeneity of caching edges. To cope with these factors, researchers propose more complex solutions. In [10], authors use branch-and-bound to transform a dual-nested problem into a MILP. In [12], the authors use Lagrangian relaxation and hierarchical prime-dual decomposition to decouple the ILP problem into two-level. In [35], and [33], authors use full polynomial time approximation methods. But these solutions are still sub-optimal at a price of complexity. For a deeper understanding, a detailed comparative analysis for content caching is tabulated in Fig. 1.1.

1.2.5 MEC-assisted caching strategies

The MEC is an emerging technology that sheds light on the 5G era with the cloud-computing capability of network edges. The survey [47] explores the benefits of MEC-based content caching by easing information processing within the cellular radio access network (RAN), without passing it to the CS. A comprehensive survey on resource management and edge computing paradigms is made in [48], [49]. In addition, the MEC helps with dynamic data rendering it helps to boost content placement and delivery. They suggest that MEC can be used to compute the location of UEs, preprocess the data, refresh the data, and predict the requests and location of users. The work in [50] shows the usefulness of MEC features while applying Dynamic Adaptive Streaming over HTTP (DASH) video caching to reduce staling, starting time, and better video quality. Many works such as [51], and [5] justify the outperforming of learning-based proactive caching strategies with the MEC excelling effectiveness.

As explained in the last paragraph of Subsection 1.1.1, the caching strategies had experienced gaps in handling network heterogeneity and modeling ineffectiveness. In addition, the cooperative caching strategies either allow content repetition or partition them. In the case of joint caching, they consider placement and delivery at the same time period and also have short of attaining updated user statistics. To address these holes, we develop comprehensive caching strategies that apply MEC-assisted data analytics, subjected to a list of constraints.

1.3 Possible Cache Edge Collaborations

Multiple caching edges that compound the cluster form a horizontal collaboration that varies based on the degree of communication freedom. These collaborations will be one of the following three schemes:

Tight collaboration

In this scheme, the caching edges have full cooperation where so that only one copy of the content is placed at only one of the collaborating MHs. This helps exploit cellular resources, their resources such as storage and computing capacity, to increase network performance [10], [11]. This scheme is more practical in the case of D2D communications. While edges share contents, the fronthaul bandwidth is not a limit, and the content transfer cost is zero; thus, content redundancy at multiple MHs is useless.

Cost-based collaboration

This scheme assumes that some resources are shared among the caching edges at unavoidable costs. That means, even though each edge can access resources from other edges, there is a recurring cost due to network limitations. However, the transfer cost is insignificant compared to the fetching cost from the CS. On the other hand, another hierarchical collaboration is established when commonly requested contents are shared freely while others are shared at a high cost.

Cost-barred collaboration

In this collaboration, the caching edges share resources at a very high cost. This implies the edges are not open for cooperation, so they probably have content redundancy. That means multiple MHs can cache content independently, and the performance gain of the network is measured for each MH. This scenario best works when the interest of each MH towards the contents is fully independent and different.

Taking the number of content sources (*e.g.*, the MBS), and destination caching edges referred to as the MHs, four vertical collaboration types are shown in Table 1.1. The first vertical collaboration is created when a single source MBS caches contents to a single MH. The second one is when a single MBS caches to multiple MHs, which is complex and gets worse based on the type of content popularity and horizontal cooperation. The third vertical is when multiple MBSs cache contents to a single MH, which is also complex due to conflict of interest among the sources such as CDNs. The fourth collaboration is where multiple MBSs cache contents to multiple MHs. This case is extraordinarily complex to solve optimally and is not covered in this dissertation.

1.4 Contributions of the Dissertation

In this dissertation, we have deeply studied content caching problems and resource optimization strategies: the modeling schemes and their solving algorithms. The main contributions of this dissertation are concisely explained in the following paragraphs.

In **Chapter 3**, a dynamic programming (DP) based content caching strategy is proposed, called 0/1-single Knapsack Problems (*DP-ZoSKP*). The DP-ZoSKP caching strategy aims to place contents from a single MBS to an MH, with different content sizes and popularity. The caching problem is efficiently modeled by the 0/1-single Knapsack Problems (ZoSKP), subjected to cache size constraint. Then, the model is optimally solved, at a pseudo-polynomial time complexity, by the proposed novel algorithm that applies dynamic programming. The DP-ZoSKP strategy is proven to outperform baseline strategies for various network parameters. This strategy is scalable to other network types thus, functions as a building block to other caching strategies. Besides, introducing combinatorics and DP to cellular optimization is a unique supplement.

In **Chapter 4**, an exact cooperative caching strategy is proposed, called bound-and-bound 0/1-Multiple Knapsack Problems (*BB-ZoMKP*). The strategy aims to cache contents at a cluster of heterogenous MHs, controlled by a central head. In this strategy, the aggregate popularity of contents is taken for each MH, and the caching problem is uniquely modeled using the 0/1-Multiple Knapsack Problems (ZoMKP). It effectively works for contents of diversified sizes, without partitioning and without overlapping over the MHs. This super complex NP-hard problem is reduced into several ZoSKP subproblems sequentially and optimally solved. Then, an advanced bound-and-bound algorithm that effectively handles the heterogeneity of network parameters is applied to solve the main problem. Each iteration of the algorithm is solved using the DP, making it more efficient than the baseline strategies. An exhaustive system-level simulation is made to examine the performance of ZoMKP, and useful design guidelines are proposed for different scenarios.

In **Chapter 5**, three innovative cooperative caching strategies are separately proposed. The strategies commonly aim at making a *demand-aware* content caching to multiple heterogenous MHs, as explained in the following paragraphs.

The first section proposes a non-partition caching strategy, called demand-aware 0/1-Separable Assignment Problems (D-ZoSAP). The D-ZoSAP is a very efficient caching strategy that applies a recursive method for the most realistic cellular network designs. This strategy effectively handles network heterogeneity where each MH has a different interest in each content. The placement problem is modeled using the binary Separable Assignment Problem (ZoSAP) by taking a series of constraints such as cache size, non-overlapping and non-partitioning content placement, and preserving the demand of each MH. For the first time in its kind, the strategy enables us to cache contents at the MHs where they are most 'demanded', answering the two interdependent questions: 'Where to cache a content?' and 'Which content/s to cache at an MH?'. This strategy profoundly outperforms at a very low computing cost.

The second section proposes an efficient and partition-based caching strategy, called the demand-aware MKP (D-MKP), after modeling the caching scheme using Special

Multiple Knapsack Problems (SMKP). It is a highly-cooperative strategy where contents are allowed to be partitioned based on their request rate from the MHs. The strategy virtually pools all available cluster cache, using standard relaxation and then optimally cache contents using the ZoSKP strategy. The D-MKP strategy applies multiple content partitioning approaches based on the number of chunks we want to produce. However, the proposed strategy focuses on the full partitioning of contents, giving the best cluster performance of fair placement across MHs and load balancing.

The third section models a comprehensive, demand-aware joint caching system using the bilevel optimization problem (BOP). The BOP formulation is a mutual extension of the ZoMKP and ZoSAP formulations. Following the modeling, a novel caching strategy, called demand-aware joint caching using BOP (DJ-BOP), is proposed. In this strategy, the nested problems of content placing and delivery are jointly optimized. This proceeding proposal marks the most realistic but complex scenario of content caching in the HCN, where two-level combinatorial decisions are made like the Stackelberg game. The leader's decision is to maximize content availability at the caching edges, and the followers' decision is to ensure minimized delivery costs.

<table>
<tr><td colspan="2" rowspan="3"></td><td colspan="4">Single MBS</td></tr>
<tr><td colspan="2">Content Partitioned</td><td colspan="2">Content not partitioned</td></tr>
<tr><td>Model</td><td>Caching Strategy</td><td>Model</td><td>Caching Strategy</td></tr>
<tr><td rowspan="2">Single MH</td><td rowspan="2">Similar content popularity (ρ_m)</td><td colspan="2" rowspan="2">-</td><td>ZoSKP</td><td>DP-ZoSKP</td></tr>
<tr><td colspan="2">Chapter-3</td></tr>
<tr><td rowspan="4">Multiple MHs</td><td rowspan="2">Similar content popularity (ρ_m)</td><td>SMKP</td><td>D-MKP</td><td>ZoMKP</td><td>BB-ZoMKP</td></tr>
<tr><td colspan="2">Section-5.2</td><td colspan="2">Chapter-4</td></tr>
<tr><td rowspan="3">Different content popularity ($\rho_{m,n}$)</td><td colspan="2" rowspan="3">-</td><td>ZoSAP</td><td>D-ZoSAP</td></tr>
<tr><td>BOP</td><td>DJ-BOP</td></tr>
<tr><td colspan="2">Chapter-5</td></tr>
</table>

TABLE 1.1: Studied vertical collaborations and proposed contributions.

1.4.1 Publications from the dissertation

The content of the dissertation is extracted from the following list of publications.

- T. M. Ayenew, D. Xenakis, L. Alonso, N. Passas and L. Merakos, "Demand-aware Cooperative Content Caching in 5G/6G Networks with MEC-enabled Edges," in IEEE Networking Letters, 2022, doi: 10.1109/LNET.2022.3192173.
- T. M. Ayenew, D. Xenakis, N. Passas and L. Merakos, "Cooperative Content Caching in MEC-Enabled Heterogeneous Cellular Networks," in IEEE Access, vol. 9, pp. 98883-98903, 2021, doi: 10.1109/ACCESS.2021.3095356.

- T. M. Ayenew, D. Xenakis, N. Passas and L. Merakos, "A Novel Content Placement Strategy for Heterogeneous Cellular Networks With Small Cells," in IEEE Networking Letters, vol. 2, no. 1, pp. 10-13, March 2020, doi: 10.1109/LNET.2019.2950990.
- T. M. Ayenew, D. Xenakis, N. Passas and L. Merakos, "Dynamic Programming Based Content Placement Strategy for 5G and Beyond Cellular Networks," 2018 IEEE 23rd International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD), 2018, pp. 1-6, doi: 10.1109/CAMAD.2018.8514960.
- S. Sánchez, T. M. Ayenew, and M. Mehrabi, "Cloud-Based Content Management for B5G Networks," in *Enabling 6G Mobile Networks*, Vol. 1, J. Rodriguez et al: Springer, Nov. 2022, pp. 501–537, doi: 10.1007/978-3-030-74648-3_15.

1.5 Organization of the Dissertation

The remainder of this dissertation has organized as follows.

Chapter 2 explains the basics of content caching for MEC-enabled HCN. It covers the main concepts of caching techniques, cache-enabling technologies, and KPIs. Apart from cellular networks, for a broader sense of understanding, the chapter gives a summary of content caching in other networks. In the end, the main challenges of content caching are expounded.

Chapter 3 details the first proposed content caching strategy. The first section presents an introduction and related works about similar schemes, followed by the proposed system design and model formulation, where the physical protocol and computing process are briefed. The fourth section explains the proposed DP-based content placement (DP-ZoSKP) strategy. The last section discusses some of the application-level numerical results and useful design guidelines.

Chapter 4 presents the second proposed caching strategy for the case of MEC-enabled multiple MHs. The first section is an introduction to the cluster-based cooperative content placement strategies, followed by the research motivation in the second section. The third section explains the studied system design, where physical cooperation and system-level MEC computing are detailed. The caching problem is modeled in the fourth section, by mapping the problem to multiple Knapsack problems (MKP) subject to constraints. The fifth section contains a step-by-step explanation of the proposed exact bound-and-bound strategy, which is evaluated in-depth using several system-level simulations, as discussed in the sixth section. Several design guidelines and key performance and cost trade-offs are discussed.

Chapter 5 explains a set of demand-based content caching strategies, in three sections. These sections explore more complex and real-network scenarios for two definite cases. Section 5.1 explains the case where contents are not partitioned during caching such that the content placement problem is modeled using the separable assignment problem (SAP). The section orderly explains the proposed powerful solving algorithm,

a content-selecting strategy called D-ZoSAP. In the end, the section makes simulation-based explanations for the reason behind the high performance of the caching strategy in 5G/6G MEC-enabled networks. Section 5.2, briefs a similar demand-preserving scenario but exceptionally with a content partitioning option. Its numerical analysis subsection discusses the service reward due to partition-based caching. In the end, Section 5.3, explains the cost-effective and end-to-end content caching scheme, as an extension to the scheme in Subsection 5.1. The section briefs joint caching system design, with complex MEC computations and a weak nested relation of cache phases, using the SAP formulation. Additionally, the content delivery utility cost is mathematically expressed. Selected simulation results and insightful design perspectives are explained in its numerical analysis subsection.

Chapter 6 concludes the dissertation and outlines future prospects. The first section summarizes the key contributions and findings of the research, with a glimpse of key analysis results. The second section gives insights into two major extension prospects to further augment the performance of proposed caching strategies.

Chapter 2

Fundamentals of Caching in Cellular Networks

In today's network, content caching is ubiquitous and a fundamental resource management technique to improve performance. For completeness, this chapter briefly discusses the basics of caching (Section 2.1), including modeling approaches and their solving algorithms (Section 2.2). In addition, some cache-enhancing technologies are explained that are strongly related to content caching (Section 2.3). It also shortlists caching scalability in other network types (Section 2.4) and concludes with a briefing on persisting challenges to content caching in HCN (Section 2.5).

2.1 Basics of Caching to Improve QoE

This section describes caching performance metrics, QoE and QoS, and its principal caching architecture with attributes such as popularity. We also describe key performance indicators (KPI) that are widely used in caching evaluations.

2.1.1 QoE and QoS concepts

The QoE is a new term of reference, broadly used in recent works, for measuring the performance of certain services from the user's perspective. It indicates the level of satisfaction users can get with a specific service. The QoE is strongly connected to the perception and experience of users, so this measurement is complex because it uses subjective measures such as users reporting their opinion to a controlled laboratory test. Poor QoE leads to the unsubscribing of dissatisfied customers from the services provided by network operators.

In contrast, the QoS is the measurable capability of a network to provide improved service over a set of network infrastructures. QoS can be parameterized by performance metrics such as link bandwidth, throughput, delay, delay variation, loss and error rates, security guarantees, reliability, audio and video confidentiality, and others. An effective QoS measurement is the backbone of an improved QoE, which is controlled and analyzed using network performance parameters such as application-layer information processing schemes and radio network access (RAN) statistics. Indirectly,

the QoE is the perceptual interpretation of the QoS from the user side. For that reason, a quantitative relationship is needed to match the natural-perception QoE and the statistically measurable QoS [52].

The QoS itself is delimited by the proficiency of two functionalities: multimedia applications, such as information coding techniques, and network automation, such as power resilience, congestion offloading, and error avoidance. But there is no unique convention on the views of QoS. For example, the International Telecommunications Union (ITU) has a user-centric view, which means that the network or application quality is contrasted towards the user side [53]. On the other hand, the Internet Engineering Task Force (IETF) has a network-centered view, which means that the quality of the service is evaluated by the efficacy of transmitting the packets to end users [54].

Rather than avoiding the complexity of subjective measures to quantify QoE, researchers propose objective quality models, most of which are based on how the human visual system perceives video signals. In addition, data-driven quality analysis improves the impact of distortions on videos to make subjective tests and gives better models which do not rely on those subjective tests [55]. Recent works are considerably advanced in estimating the QoE for new constraints of future HCN [56].

In summary, the research focuses on two interdependent techniques to improve user QoE: quality-based content's chunk representation and content caching. In the first method, video contents are chopped into chunks and stored with different resolutions. Each resolution gives different quality levels where the user can download its preference, based on the network condition, and maximize the playback effectiveness. A good example of this method is video streaming using the DASH video standard [57]. In the second method, complementing the first method, various content caching methods reduce the stalling duration and frequency, suppress the video quality variation, and improve video quality by maximizing channel utilization [58].

2.1.2 Content popularity

Content popularity refers to the degree of a request to be forwarded to particular content. It always takes higher attention in network optimization, where popularity-based content caching is widely used. After the analytical cache operation, a performance evaluation can be done using important performance metrics.

Content popularity can be mathematically estimated from the user request rate, every content receives from UEs, within a specific period of time [59]. The popularity of content is usually represented by the well-known Zipf distribution; however, not the only model. All contents are ranked based on their requested profile to estimate their popularity. Denoting any m^{th} ranked content by f_m, in a superset of contents $\mathcal{M}$ ($m \in \mathcal{M}$), their individual popularity (ρ_m) is given by:

$$\rho_m = \frac{m^{-\gamma}}{\sum_{j=1}^{|\mathcal{M}|} j^{-\gamma}} \tag{2.1}$$

where j is rank of contents ($j = 1, 2, \cdots, m, \cdots, |\mathcal{M}|$) and $\gamma \geq 0$ is the Zipf parameter, also called *skewness index* of the distribution [29]. This parameter indicates the skewness regarding the recorded content request rates. The γ values close to zero indicate uniform content requests while the higher value, such as γ=2, shows that only a small number of contents account for the majority of the popularity sum.

The caching policy is deterministic when content popularity is known before caching and static. However, the content popularity is an uncertain value over a period of time [60]. For that matter, several caching policies have addressed this dynamic behavior, such as content popularity and user location, using learning algorithms. These algorithms take input of content profile, from RAN information, such as request rate to specific CDN website, and predict the popularity of contents [5].

2.1.3 Caching architecture

From the caching point of view, the mobile network has two main parts: the core network (CN) and the radio access network (RAN). The caching procedure mainly concerns where to cache contents and which device must deliver the content. Since mobile users tend to access similar content repeatedly, caching it in the cellular network can be done at either of the two parts- with different features.

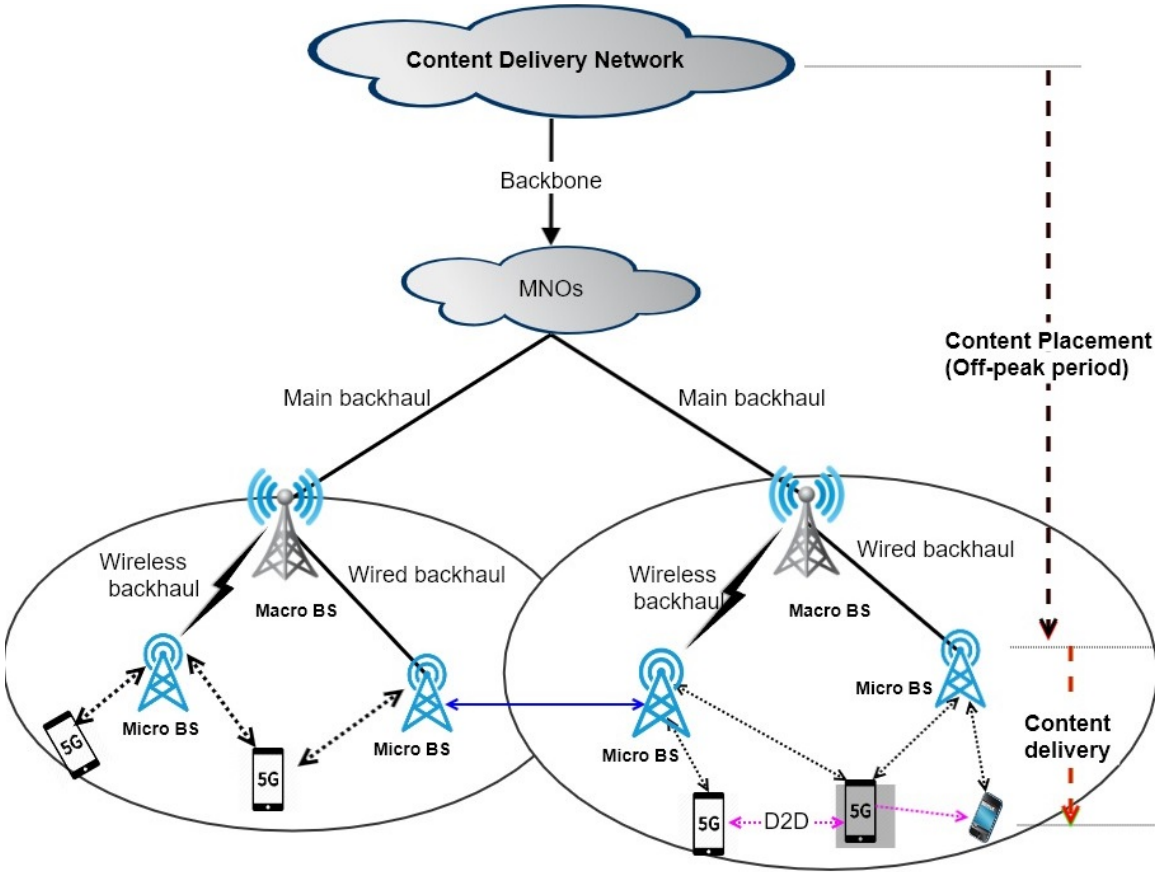

FIGURE 2.1: Simplified representation of a content caching system.

Fig. 2.1 shows a simplified caching architecture in the RAN, where cache-enabled devices, including the central MBS, are employed at different layers of the HCN hierarchy. For the caching purpose, a list of mobile and fixed edge devices such as sub-base stations (SBS), femto-base stations (FBS), and smart UEs are deployed, mainly in the RAN. These devices are deployed in different tiers of cooperation modalities, where their local distribution in the RAN is usually modeled independent Poisson point process (PPP) [32]. The caching devices are becoming computationally strong to process network information and have large enough storage to cache contents [3]. Once the contents are brought closer to users, they can be delivered to UEs at a low transmission cost and latency. For better insight, a comparative summary of the caching architectures is tabulated in Fig. 2.2, and system parameters are tabulated in Fig. 2.3.

The content caching framework in cellular networks has two distinct phases: content placement and content delivery. These steps are interdependent such that optimizing the placement directly determines the performance of the delivery process while optimizing the delivery affects where contents should be placed. This dependency makes the caching process a complex nested-dual problem [10]. In the middle of these phases, the content transmission technologies such as multipath [61] and multicast[62] play a significant role in the QoE. Over time, a content update or cache refreshment is taken as a compliment. In addition, the channel allocation and frequency usage also greatly contribute to the content delivery performance by eliminating interference in the RAN [63] or within a D2D communication[64].

Content placement

This process includes content screening, mostly based on their popularity level, and pre-fetching of contents into convenient caching devices. Alternatively, we call these devices cache-enabled cellular network edges (CNE) or mobile helpers (MH). They execute some tasks such as relaying contents from the CS to the UEs, gathering network information from the RAN, and making data analytics.

Content placement is often done during the off-peak time period; for example, at night when the backhaul network is less congested. This proactive measure helps reduce the stress on the network due to the simultaneous downloading of massive and big contents. However, in the case of time-critical services, the placement could be done as soon as the user requests. Having a prioritization among the two options becomes a complement to congestion management and cost optimization.

An efficient content placement strategy helps to improve the QoE, with constrained and non-flexible network infrastructure. The strategies are of many types based on aspects such as the level of cooperation among MHs, dynamics of content popularity, the interdependence of placement and delivery phases, mobility of MHs, and the information coding process. Henceforth, the placement strategies vary by the tools used to model the problem and the algorithms to solve them.

System model components			[49]. Song et al.	[50]. Jia et al.	[13]. Altieri et al.	[29]. Cui et al.	[16]. Fan et al.	[14]. Yang et al.	[51]. Wen et al.	[17]. Chen et al.	[52]. Afshang et al.	[6]. Golrezaei et al.	[53]. Carlsson et al.
Network Model	Number of tiers	k-tier (not listed)		✓	✓		✓		✓	✓	✓	✓	✓
		two-tier				✓							
		three-tier						✓					
	HetNet Cache edges	D2D overlay	✓		✓		✓				✓	✓	
		Small cell			✓	✓				✓	✓	✓	
		Macro Cell			✓	✓	✓					✓	
		Relay link		✓			✓	✓			✓		
	Type of mobile helper	Macro BS (MBS)		✓		✓	✓	✓	✓			✓	
		Micro BS (SBS)		✓		✓			✓	✓	✓	✓	
		UE (D2D link)	✓	✓	✓		✓	✓		✓		✓	
Traffic	Multimedia content		✓	✓		✓		✓					
	Video				✓		✓		✓	✓		✓	✓
Content Transmission	Channel property	Path loss	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	
		Rayleigh fading	✓	✓	✓	✓	✓	✓		✓	✓		
		Interference	✓	✓	✓		✓	✓	✓	✓	✓		
		Noise (additive)		✓			✓	✓					
	Modulation technique	Orthogonal						✓		✓		✓	
		TDMA			✓		✓					✓	
	Spectrum assignment	Separate spectr.			✓		✓						
		Same spectrum	✓					✓				✓	
Cache Edge Distribution	Macrocells	I-HPPP		✓	✓	✓	✓	✓	✓				
	Microcells	I-HPPP				✓			✓	✓			
		Clustered									✓		
	UE distribution	I-HPPP	✓	✓	✓	✓	✓	✓	✓	✓			
		Clustered									✓	✓	
	Mobile Helpers	I-HPPP	✓	✓				✓	✓	✓	✓		
		Fixed										✓	
	Relays	I-HPPP								✓	✓		

FIGURE 2.2: Comparative analysis of cache system designs.

System model parameters			[49]. Song et al.	[50]. Jia et al.	[13]. Altieri et al.	[29]. Cui et al.	[16]. Fan et al.	[14]. Yang et al.	[51]. Wen et al.	[17]. Chen et al.	[52] Afshang et al.	[6]. Golrezaei et al.	[53]. Carlsson et al.
Coordination technique	Clustering metric	Cooperation distance		✓	✓	✓	✓	✓		✓		✓	
		URL & Content index		✓					✓		✓		
	Cluster shape	Discs with radius				✓						✓	
		Hexagonal grid								✓			
		Dynamic					✓	✓			✓		
Content Placement	Popularity distribution	Zipf	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	
		Ephemeral Zipf											✓
	Cache size	Fixed, finite, library	✓		✓	✓		✓	✓	✓		✓	
		Fixed cache size	✓		✓		✓	✓	✓	✓		✓	✓
		Fixed, equal files size	✓			✓	✓	✓		✓	✓	✓	
		Flexible file size											✓
	Content Assignment	Random	✓		✓								
		Controlled		✓					✓	✓	✓	✓	
		Hybrid				✓	✓			✓			
Delivery	Through a D2D link		✓	✓	✓			✓			✓	✓	
	Macro BS (MBS)							✓				✓	
	Non-caching UE							✓			✓	✓	
	Cache-enabled UE							✓			✓		
Other	MH-to-UE	Deterministic assignment					✓	✓			✓	✓	
		Random	✓		✓	✓			✓				
	User requests modelled				✓	✓		✓	✓		✓	✓	✓

FIGURE 2.3: Comparative of system parameter assumptions.

Content delivery

This phase refers to the techniques used to serve cached contents from MHs to UEs. It includes the transmission process and decoding of requested content by the receiver side. Devising an efficient delivery strategy improves the downloading latency over cache memories. Hence, it depends on the technology we deploy, such as the transmission rate and the assigned spectrum. Especially the time-related human perceptions and most of the QoE-related indicators, such as delay and jitter, are impacted by the content transmission technique. However, valid interpretations are not yet fully defined to know the optimal transmission techniques [65].

When a request is made by a UE to an associated MH, the content is transferred to the UE if found in the MH library. Otherwise, the search for contents extended to other MHs or the CS. Whichever edge has the content, passes to the UE based on the cooperation protocols of the delivery. In fact, only if the content is not found in a cluster will it be served from the CS. In some cases, the request comes from the caching device itself so that the response is immediate.

The content delivery phase depends on physical layer-related factors such as power consumption. As such energy-efficient delivery strategies reduce the costs of high data rates and distributed caching in HCN, which improves the QoE [66]. This process is also affected by information-level processing algorithms, *i.e.*, encoding and decoding schemes. Some delivery techniques use coded caching [67] while others apply uncoded delivery schemes [35].

2.1.4 Key performance indicators (KPI)

Key performance indicators are metrics used to evaluate the performance of network-specific optimization techniques such as content caching. Setting up a relevant KPI needs to address the difficulties of the chosen scenario and applied technology. In this subsection, we explain some KPIs linked to content caching, defined by standardizing institutes and research initiatives such as the 5G Infrastructure Public Private Partnership (5GPPP), the 3G Partnership Project (3GPP), and IETF.

Playback experience

The playback experience is conceptualized using the QoE and is inspected by the bit rate loss of the transmission. This metric is complex and depends on other parameters such as video download start-up time, buffering level, and video resolution. The buffering plays a key role in resuming the anticipated video quality when users play unfinished content while simultaneously downloading the remaining part [68]. Experimental improvements have been developed using content placement techniques for mobile users in web services like Netflix, Amazon, and YouTube, where content placement and delivery are the basis of their efficiency. In urban crowded scenarios, this technique is efficient. However, it fails when the content cache hit ratio is degraded.

Throughput

The network throughput is the amount of the relevant data that is successfully transmitted from a sender and output at the receiver, using the cellular connection. Its performance depends on the applied delivery techniques and is typically measured in bits per second (bps). Throughput is a critical system design parameter, which highly influences the user experience towards a service [32]. From this perspective, the main optimization objectives are designing throughput fairness among competing UEs or maximizing throughput [69]. Several closed-form expressions are developed for various network scenarios [70].

Response delay

This performance metric shows the service latency from the content request until the user gets the requested content. The network delay strongly affects the QoE and very essential in the case of time-critical services [71]. The delay in the RAN varies depending on the mobile devices, where the round trip time (RTT) latency will drop to 1 ms for the fifth generation cellular (5G) and beyond networks [72]. The response delay includes three time periods: the period used to process the content at the CS, the period required for queuing, and the one spent for transmitting to UE. Various strategies are proposed to reduce these time segments. For example, content caching eliminates the first two time segments and reduces the third. In a more advanced case, applying the MEC-capable MH in large density helps to significantly reduce the latency by offloading the network congestion [73].

Area spectral efficiency

This metric is traditionally linked with the experienced user data rate; however, a concrete definition is not yet set. Most often, area spectral efficiency (ASE) refers to the maximum data rate that a specific bandwidth unit can support in a given period of time and coverage area, with a unit of $bps/(Hz.m^2)$. It indicates the efficiency of spectral utilization and transmission power over a link [74].

Content providers and network operators are experiencing difficulties in maximizing the ASE with physical resources. New spectrum bands have been considered to improve network resource management in high data rate scenarios. Careful design perspectives, such as trading off MH cache capacity to MH density and popularity-based caching strategy, help maximize the ASE in cellular networks [75].

Energy efficiency

Energy efficiency (EE) quantifies the power consumption rate in the network infrastructure. Shortly, it is the ratio of the total transferred bits to the total power consumption [76]. The EE is a widely used metric, predominantly expressed in the bit-per-Joule unit, to evaluate the energy consumption for the 4G-LTE and beyond cellular systems. For example, the network system that can deliver higher data at lower energy consumption is efficient for a given frequency bandwidth, and time-period [72].

The EE includes the power consumption of the core system, the network edges in RAN, and UEs, which means the power bill of the mobile operator and the user. Hence, the network design considers optimizing all these energy consumption components so that the energy efficiency is maximized and achieves green communication [77]. The EE maximization is done by modeling the power consumption in closed form and solving it using various algorithms, such as approximations, while applying content caching technology. For example, eliminating transmission redundancy of contents reduces the energy consumption at the CS and the MHs, hence improving EE [78]. In another case, devising an optimal queuing scheme while multicasting to multiple MHs enhances the number of users served and the EE [79].

Cache hit probability

The cache hit probability (CHP) shows the availability of contents in the RAN and conceptualizes the necessity of content placement. This performance parameter is defined as the probability that a requested content, by a user, can be found stored at an eligible cluster of MHs [29]. From the end-to-end view, it is also defined as the probability of content being successfully delivered to the requesting UE [33]. From a set of contents $\mathcal{M}$ in the CS, where the popularity of each content f_m is denoted by ρ_m, the simplified CHP of cached contents is expressed as:

$$\Psi_{\mathcal{M}} = \sum_{m=1}^{|\mathcal{M}|} \rho_m \cdot x_m \tag{2.2}$$

where x_m is the content placement indicator. In more complex closed-form expression, the CHP depends on many factors such as physical-layer parameters, content-level parameters, and type of MH cooperation [75].

Coverage probability

Coverage probability is the probability that a random request from UEs is successfully served by the associated MH in the network. It is estimated by comparing the attained signal-to-interference ratio (SIR) to a threshold of the required transmission rate by the user [80]. Coverage probability also indicates the successful decoding of content at the end user [29]. Researches show that the coverage probability increases with the deployment of advanced techniques such as contention-based controls, content popularity-based caching, cooperative caching, and clustering. The coverage probability depends on many factors, such as the content popularity, the MH density, the required transmission rate, cache sizes, and other physical-layer parameters [39].

Cache hit ratio

The cache hit ratio (CHR) parameter shows how many content requests can a caching system fulfill successfully. It is broadly used to measure the performance of any cache scenarios. A cache hit happens when the requested content is fulfilled by the cache

instead of the remote server, the CS [68]. Hence, the CHR is calculated by dividing the number of the occurred cache hit events over the total number of requests [81], sometimes expressed in percentages for some network types such as CDNs [82].

The CHR indicates the mass and load distribution of contents arriving at the caching system [14]. It is also considered a 'reward function' that indicates the instant traffic demand while downloading a file from a server [30]. Due to such broad considerations, CHR is a more general performance measure parameter. It is formulated for each placement strategy and specific scenarios, where also complex utility functions can be characterized using it. Most often, the CHR value (Φ) of a specific caching strategy § is expressed as:

$$\Phi(\S) = \frac{1}{|\mathcal{U}|} \sum_{u=1}^{|\mathcal{U}|} \sum_{m=1}^{|\mathcal{M}|} r_m \cdot h_{um}(\S) \tag{2.3}$$

where $\mathcal{U}$ is set of users, r_m is request rate or probability, and $h_{u,m}(\S)$ is the probability that user u can be satisfied by requested content f_m under placement strategy § [83].

2.2 Caching Models and Solving Tools

Mathematically, many caching techniques have been proposed in the area of network optimization. Setting aside many network types, we focus on content caching strategies applied to only cellular networks. Yet, caching in cellular networks is a complex process, and a wide range of modeling and solving tools are used [32].

While designing caching for HCN, we consider a list of parameters and their dynamic variations that can be evaluated numerically or analytically. The most common input parameters to treat are content popularity, transmission rates, content size, cooperation distance, SIR, and coding rate. Having the parameters, the aim is to meet the QoE, such as traffic offloading, service delay, energy consumption and efficiency, area spectral efficiency, network edge densification, cache hit probability, utility cost, and others. However, the HCN network is non-flexible and with a bunch of constraints to deal such as network bandwidth, cache size, and channel capacity [7], [84].

Based on the type and nature of caching models, convenient solving algorithms are suggested, such as random, DP, Linear programming, MILNP, Hungarian algorithm, greedy and approximations, heuristics, genetic algorithms, and others [32]. The caching models can be categorized into three main types, as described below.

2.2.1 Stochastic geometry

It is a statistical method used to model and design cellular network features such as content caching. Many authors adopted such probabilistic and stochastic expressions to give detailed intuitive models for content caching, not only for large-scale HCN designs [11], [14], [24], [85]. It is widely used for effective resource allocation and improving content caching gains.

The stochastic expressions are tractable and strong enough to capture the dynamic behavior of caching environments but always depend on strong assumptions on critical cache system attributes, such as equal content sizes, uniform popularity, and static content location models [86]. These assumptions are mainly taken to reduce the complexity of analytical stochastic expressions and solutions.

2.2.2 Game theory

This is a mathematical tool where UEs or MHs as players compete for resources, such as cache space, to maximize their objective functions by rational decisions [87]. While maximizing the user QoE using caching, the competing parties prioritize increasing their benefits and should carefully limit their costs, such as energy consumption and cache size. The system achieves the Nash Equilibrium when non of the players can individually maximize its profit. The game theory approach is practically essential for the case of proactive caching, where the playing devices extensively interact [44], [45].

There are many game types to model caching process, both centralized and distributed, with various scenarios in the HCN. For example, if we apply the centralized caching perspective and when the players are the ISPs, to cache their contents on MHs and effectively share the limited cache size, we can model and solve it using *auction game* [49]. In another case, when the caching helpers in the RAN are movable MHs, within the predicted route and maintain the pricing fairness for buffered QoS levels, we can apply the *contract game*. Differently, suppose we apply a decentralized caching system such as on D2D communication, where the devices form a cost-effective and cooperative caching cluster. In that case, we can use the *coalition game* [88], [89]. In a similar but more complex scenario, if we choose to cache contents at moving edges that use the vehicle-to-vehicle (V2V) protocol, we can apply *evolutionary game* [44].

2.2.3 Optimization

In this category of content caching model (convex and non-convex optimization or a range of heuristics), various techniques are included, where their common goal is using optimization algorithms to solve the model. Combinatorial optimization, linear programming, non-linear programming, Lyapunov optimization, index coding, and general assignment problems are some of them [11].

Based on the models, the optimization problems are solved either theoretically with closed-form expressions or heuristic methods. To that extent, many solutions are commonly used, such as greedy, relaxation, approximation, branch and bound, subgradient optimization, Karush–Kuhn–Tucker (KKT) method, and random selection algorithms. Each of these approaches does have several ways of applying techniques. Many proprietary and open-access standard optimization solvers also deploy advanced versions of these algorithms.

Others use complex placement matrices to optimally solve the content caching by modeling it by convex functions so that it is numerically solved using standard convex-optimization solvers [90]. In this work, the single-tier HCN is a convex function solved

by setting the derivative of its Lagrangian expression to zero, which gives the optimal Lagrange multiplier. However, for the multi-tier case, the optimal content placement probability is done by approximating the hit probability such that the effect of one tier is neglected on the other.

2.3 Cache Enhancing Technologies

As a promising technology of network performance enhancement, content caching itself can be underpinned by advanced technologies such as massive machine type communication (mMTC), D2D communication, popularity prediction strategies, multipath, MEC, and cache replacement, as discussed below.

2.3.1 Massive machine type communication

The 5G system is orchestrated to bring a wide range of communication services and applications under the same umbrella, from individual mobile applications to smart cities. These applications and services are shown in the triangle of Fig. 2.4, whose three extreme edges indicate three pillars: i) Enhanced mobile broadband, ii) Massive machine type communication (mMTC), and iii) Ultra-reliable and low latency communication (URLLC) [72]. The ITU-T Study Group 13 proposes that future data-aware networking (information-centric) will achieve 5G ultra-low latency by enabling proactive in-network data caching and limiting redundant traffic in core networks.

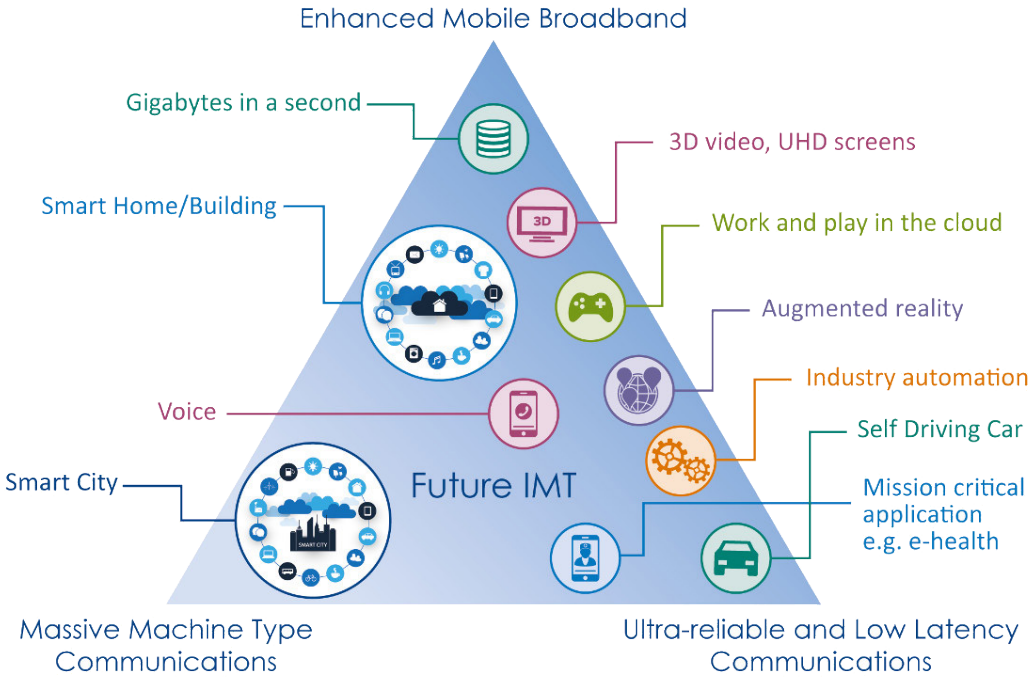

FIGURE 2.4: Some of outlined 5G use cases [72].

Machine-type communication incorporates a massive number of smart devices, which are resource constrained. The devices have different communication layers and protocols. This mMTC technology can be effectively used to enhance content caching performance. A robust deployment and cooperation of these intelligent devices mitigates several 5G network constraints. For example, having a border router, such as ZigBee, enables one to connect devices in an mMTC environment. Then, a smart home with devices connected to the core network via a border router can cache the most required contents. It also fosters high-speed video streaming of different resolution levels and QoE requirements.

2.3.2 D2D communication

The HCN's communication includes a device-to-device (D2D) communication where a cache-enabled user device serves a content request from another co-located device in its proximity. While using the D2D communication, one user may not ask for content from the backhaul network but easily takes it from another neighboring device [20], [21]. This collaboration can bolster content caching in the mMTC environment [91]. This protocol helps offload content to the RAN, reducing the traffic during peak times. However, it is hard to guarantee user privacy and fewer tractability challenges.

While applying the D2D in a cellular network, we must take care of the spatial distribution of the devices and optimize the distances to maximize caching gain. A family of admissible protocols by which users are spatially clustered based on their distance is used, such as the type-II Matern hard-core Poisson process (PP), and the translated PP [15]. These protocols optimize users' distance in a stochastic cluster to reduce the attenuation and improve performance. With such analysis, a substantial number of requests can be served through a distributed caching strategy through the D2D without having a dedicated caching infrastructure such as CDN.

The D2D is applied for low-latency and high-data-rate demanding applications. The locally cached content is served within the communication link of smart devices in the ecosystem without the direct involvement of the BS. Hence, the D2D link reduces the network response time. Since there is no need to fetch the content from the backhaul for every request, it minimizes the backhaul congestion. The D2D also reduces the redundant demand of contents to the backhaul network. This reduction avoids the computational consumption for every request. This process improves the frequency reuse, energy efficiency, and coverage extension and reduces backhaul load [18], [29]. However, D2D has unresolved drawbacks, such as high signal wastage, higher battery drains, cross-interference, and optimal spectral efficiency.

2.3.3 Learning strategies

Many learning strategies are used to enable dynamic caching based on predictive approaches to user statistics and content attribute predictions. In dynamic caching, the bulk of user information is collected by network servers and computed by the MEC to learn about the network's dynamic behavior. Many predictive caching models use machine learning (ML), deep learning, entropy-based, and other learning strategies.

The critical role played by ML comes with the idea of learning from the network information to make optimal caching decisions. For instance, many works propose ML-based caching strategies to increase the mobile network operator revenue [92], [93], by using the machine learning methods such as the Markov decision process to reach the best set of feasible solutions. Also, some proactive and reactive proposals have been proposed based on reinforcement learning algorithms [51].

In learning-based caching, the type of contents, location, and edge network are predicted based on statistical information. To that end, the MEC system can take real-time network information and make cost-effective big data analytics to decide to route and control the content assignment. For example, many relevant works propose the implementation of clustering and regression techniques to predict the position of the users [94]. However, predictive caching needs huge data and a long-time learning process. It also needs highly correlated network data, which is the constraint of the scheme that most network variables, such as content popularity and user location, are stochastic. In the same way, the high-data dimensional and sparsity, such as hyperspectral multimedia contents, are very complex to train.

2.3.4 Multi-access edge computing (MEC)

With the integration of multi-access edge computing (MEC) capabilities in 5G mobile cellular networks, mobile network operators can place popular video content closer to the network edge. They can do this at off-peak times by predicting user requests that exhibit a high correlation over smaller geographical regions for a given time interval.

The MEC is an emerging technology that sheds light on the 5G era with the cloud-computing capability of network edges. This protocol brings the centralized content processing task into distributed network edges or MHs, with a solicited decoding capability. Hence, these MHs can run applications and information within the cellular radio access network without passing it to the central server. Comprehensive surveys on resource management and edge computing paradigms are made in [48], [49].

While benefiting from MEC capabilities, it helps meet ultra-low latency requirements, real-time processing, location awareness, and personalized high QoE to a large crowd of users [95]. Some MEC platforms also solve the requirements for accelerated content delivery, and timely data usage [96]. In addition, it helps dynamic data rendering that boosts the placement process. This data processing is vital for content delivery, as well, because the network edges can take a share of expensive preprocessing during the transfer of content[51], and optimizes based on given information from the user radio interface. More technically, the MEC features are helpful to have reduced staling, faster starting time, and better video quality [50].

The MEC helps to handle mission-critical applications, value chains, and real-time network services such as augmented reality, intelligent video acceleration, connected cars, and IoT. In other words, mobile caching is a fundamental technology for these critical services, influenced by the MEC technology for the service to be maximally effective. Since MEC is close to the user, it reduces the end-to-end computation time. That means it reduces the response delay and relieves the network congestion [96]. For

example, the newly emerging augmented reality needs a high data rate and low latency content delivery. Hence, MEC can be used to compute the user's location, preprocess and refresh the data, and predict the requests and location of users [51].

2.3.5 Cache refreshment techniques

To avoid the overflow of cached contents and the fact that not all files are kept in the cache forever, the caching strategy needs an efficient and real-time replacement or refreshment algorithm, where less popular or contents whose popularity period has expired have to be removed from the memory stack [3].

There are several cache replacement strategies proposed in the state-of-the-art. Some works focus on the data-structure perspective, such as first-in-first-out (FIFO) replacement, where the first cached content will be evicted from the cache. This approach is the easiest and most popular strategy; however blind to the future demand of the content [97]. Alternatively, the frequency-based FIFO replacement method is used, where variable-sized content segments are stored in the cached, and the FIFO strategy is applied based on usage pattern [98]. The other cache refreshment categories are content popularity-based dynamics, such as the aging popularity-based cache replacement strategy that combines the timing and updated popularity statics. In this case, old and unpopular files will be evicted from the cache [99]. Separately, some techniques rely on the least frequently used (LFU) content replacement approach [97] and the least recently used (LRU) content replacement technique [100].

2.3.6 Multipath protocols

Multipath caching is where contents are placed into caching edges through multiple traditional protocols, such as the user datagram protocol (UDP) and transport control protocol (TCP). This is approach is the adoption of multipath data transport protocols such as *multipath transport control protocols* (MPTCP) and *multipath transport quick-UDP Internet Connection* (MP-QUIC) [101]. A few studies showed that multipath protocols enhance sustainable communication and reduce segment loss; hence, it presumably enables caching across cellular networks. Especially this mechanism is useful for *content showering*, instantly storing traffic contents to a bulk of users, such as in a stadium.

The multipath protocols for cellular networks bring the capability of regular TCP between multiple caching hosts. In doing so, the multipath protocols help us access one network node (resource) via multiple paths over the Internet or private networks. In this scheme, the contents can be synthesized as frames appropriately at the source, scheduled to transmit through various paths, and reordered at the receiver side. This gives considerable gains in increasing network performance, but each path option has different path characteristics, and there should be an optimization technique.

What matters for the performance of this multipath caching is how the transmitter manages the content, such as segmenting, framing, packeting, and stream forming from the CDN. Thus, information-aware and optimized content management help achieve the required QoE. When a request (to any chunk) is received from UEs, the

receiving server creates a number of coefficients that are linearly independent of the code in the request. After having the coefficient, the server encodes the chunks to the same number of blocks. Then, the server selects the block and transmits it with an appropriate interest forwarding scheme.

In the current HCN, creating a multipath protocol is a viable option due to the convergence of several network types, such as 3G UMTS, 4G LTE, 5G NR, WiFi, and MiFi, into the 5G NR architecture. These networks serve as alternative paths where the content chunks can be transmitted. However, integrating these heterogeneous networks becomes a fatal threat because they have different protocols, leading to inefficient handover and lossy connectivity. Standardization bodies have reached a substantial milestone in devising advanced techniques that achieve seamless integration, such as *dual connectivity*, non-3GPP Interworking function (N3IWF), and access traffic steering switching and splitting (ATSSS).

2.4 Caching in Other Networks

Content caching strategies are also employed in other networks to maintain their QoE–like cellular networks. In this section, we shortlist some of those network types for better insight into caching strategy scopes and envision the technological scalability.

Caching in IoT networks

It is well anticipated that the IoT network will have a low traffic rate. Still, due to billions of networked things over the entire Internet, its aggregated traffic data creates a significant load on the backhaul network [102]. However, note that the heterogeneous IoT data traffic behaves quite differently than the traffic in cellular networks. The IoT network deploys devices that use low-power, low-rate communication and produce limited packet size. In addition, the IoT data is transient, and its content popularity does not follow widely accepted distributions as they have a short stay. This means a unique caching strategy is needed to reduce network traffic, maintain content freshness, and reduce energy consumption [103].

The research on IoT caching has been nearly a decade where, in most cases, cache-enabled IoT devices and dynamic routers are used as caching helpers [104]. These helpers take the trade-off between multi-hop traffic load and data freshness, which users need to have [105]. Recently, the information-centric network (ICN) architecture for caching in IoT networks is becoming dominant [106]. In this scheme, contents are uniquely identified by named data networking (NDN) and stored in a distributed manner. This naming makes data access easy, scalable, energy-efficient, and highly mobile. The NDN is very effective for IoT caching, and MEC-enabler towards reducing retrieval latency [102]. In addition, the ICN approach helps guarantee secure data communication, one of the most challenging limitations of caching.

Caching in vehicular communication

Due to the recent development of mobile operating systems that mirror mobile applications to the car dashboards, such as Android Auto and Apple CarPlay, vehicles' surplus content demand is observed. For example, smartphones can mirror video games and resource-demanding applications on car screens. Thus, vehicular communication is becoming a potential caching protocol where medium-level contents can be fully distributed and delivered to passing-by users [107]. By exploiting vehicle-to-vehicle communication, contents can be exchanged among their cache-limited edges.

In this platform, the caching edges at roadsides are supposed to cooperate with the onboard vehicular content caching edges, thus offloading the backhaul. Many vehicular network frameworks have been proposed, such as crowdsourced vehicular content-centric networking, WiFi-based moving, parked, and coalition-based vehicles [107]. This proliferation of vehicles is forming an Internet of vehicles with different velocities, where the traditional TCP/IP protocols need to maintain scalability and reliable addressing of moving vehicles. Instead, the ICN networks are getting better attention to curve these dynamic natures, among which the newly dominating NDN technology is much preferable. This approach helps users to request the content using its unique identifier without worrying about the source and destination IPs and content source edge. This new technology provides secure, reliable, and robust communication among the vehicles that are supposed to enhance onboard caching [108].

Caching in satellite communications

Currently, several Internet satellite constellations such as Telesat, OneWeb, and Starlink are being built, which is a promising platform to deliver broadband connectivity to inaccessible areas. In advance, the current 5G wireless and the future 6G technologies must seamlessly integrate terrestrial-satellite communication systems. However, it has a large coverage area and high data rate, and the long propagation delay (whose round-trip delay is up to 270 ms) is the gridlock to the QoE of the satellite communication, which can be enhanced using distributed content caching [109].

Unlike in the conventional terrestrial-satellite networks, where the satellites serve as backhauls and contents are placed only at the main base stations, the integrated-satellite network can cache at edges such as MBS, satellite, and gateway that cooperate to serve users at ground [110]. In some cases, load-balancing schemes protect overflow due to the limited storage by exploiting the vast cache resource in multi-layered satellite constellation [111]. Applying the MEC and storage pooling, both at the satellites and terrestrial edges, helps to reduce service delays [110]. Recent works also show that, by taking advantage of three-layer cooperation, the hybrid satellite-aerial-terrestrial communication is a disruptive strategy to maintain enhanced mobile broadband and also confining construction costs [112].

Caching in Ad-hoc networks

Mobile caching for ad-hock networks, where all the nodes in the wireless network can serve as cache-routing edges and UEs, has taken some attention from the research community. However, the characteristics of this network face hurdles, such as the caching sizes of the edges being minimal, so it needs a centrally controlled optimization mechanism. However, unlike other networks, the active ad-hoc edges are self-organizing and join the network randomly, but the edge association fluctuates quickly. These properties cause the network dynamics to be relatively high so caching for ad-hoc networks becomes more complex than conventional networks. To manage secured and reliable caching in the dynamic nature of the ad-hoc, content-centric networks are preferable [113]. Among them, the NDN technology is promising in that it can keep cached contents longer, regardless of caching edge loss from the connection [114].

2.5 Challenges of Content Caching

Though the literature on content caching in cellular networks counts decades back, due to its dire complexity and challenge, it continues to attract the attention of researchers. This section explains some of the tempting challenges.

2.5.1 Content popularity variation

The vast majority of the proposed caching strategies are popularity-based. Still, not all contents are cached, such as real-time applications and control signals, as users do not frequently retrieve them. These contents have nearly zero popularity but are active in the network, creating contention among MNO and CDNs. More severe than that, the popularity of contents encounters frequent spatial-temporal variation [115]. As a result, neither static nor dynamic modeling tools, such as learning and predictive approaches, can effectively characterize content popularity. Even with suppressive assumptions, the caching problem becomes NP-hard. These setbacks unfold the ineffectiveness of existing content caching strategies. On top of that, the popularity of content greatly varies among users, which forces researchers to shift to clustering mechanisms while the optimization problem is practically unsolvable.

2.5.2 Edges' mobility

Mobility is one of the main challenges in cellular networks since UEs and some MHs are non-fixed. Mobility of these edges causes a sudden change in popularity and connection [116]. Both causes, popularity disruption and mobility of entities lead to tough challenges in caching. It highly affects content placement by impacting selection and routing processes. Mobility also defects content delivery by causing a high cache miss rate, network topology variation, and route changes.

Several studies have attested to find tractable expressions for device mobility, such as stochastic geometry models for realistic scenarios [116], [117] and machine learning techniques for patterning big data generated by users' mobility [118]. In the first model, UEs are treated with parameters such as arrival time, departure, and pause inside a specific region. However, it comes out with highly complex mobility patterns. In a more complex way, some techniques apply dynamic modifications to trace the UEs for their trajectories along known and distributed hot spots. Other modeling schemes use machine learning algorithms to train the network and capture UE mobility correlation. So, they can predict a set of key mobility parameters: the velocity of UEs and their pause periods, and their usage behavior when they regularly visit specific clusters [118]. But, these mobility tracking techniques need to be revised to alleviate the challenges caused by mobile network entities.

2.5.3 Limited capacity and storage space

Contrary to the extremely high data rate and content proliferation, the backhaul capacity and storage space are limited in the HCN and remain bottlenecks for the 5G performance. Thanks to disruptive technologies that improve the network ASE and allocation of wide spectrum chunks at high-frequency bands, along with network densification in the RAN, they have brought the capacity of the backhaul and fronthaul links to their capacity limits. From a caching perspective, this maximum-capacity achievement is sufficient because the process happens at an off-peak time. However, the channel capacity gets exhausted at peak times and remains a cache constraint. Much worse, storage resources are limited and frequently run off while caching.

Conversely, caching is proposed to offload backhaul capacity, using several optimization techniques to exploit these two constraints. For real networks, many links exist with different bandwidths and storage capacities. When we try to establish cooperative caching without content duplication in a cluster, it ends up with an exponential number of constraints and is quite hard to solve optimally.

2.5.4 Heterogeneity of the network

As its name indicates, the current 5G heterogeneous cellular network experiences many diversities regarding protocol types, network attributes, and service requirements. From the attributes perspective, the 5G new radio (NR) will host real network properties such as different content sizes and cache spaces, varying popularity towards each MH, and different bandwidth and computational capacity. As such, content caching in multi-layered and heterogeneous architecture poses an unarguable challenge that limits the degree of freedom: where to cache, which size of objects and storage to use, and which connection capacity.

From a protocols perspective, the anticipated 5G accompanies multiple 3GPP and non-3GPP network types, which have different frequencies and standardizations. Having such heterogeneity, it is yet unclear how to create seamless integration, where apparently, multiple path connections needed to be established using these network

types. Although having multipath is an opportunity to enhance the content caching process, the uncertain integration of heterogeneous networks becomes highly problematic for the placement and delivery phases. That means a sudden change in the placement and delivery paths incurs a service cost and a cache miss. A few techniques are under experimentation to seamlessly integrate these technologies into one umbrella of the 5G NR without hardware changes, featured in Subsection 6.2.1.

2.5.5 Computational limitation

The computationally demanding caching solutions are implemented in parallel with real-time operations, while the computational consumption is high for selection algorithms and content preprocessing to enhance the QoE. Hence, the services such as adaptive video streaming, online gaming, telemedicine, IoT applications, and other mission-critical operations scramble with the caching process.

Distributed caching strategies need to make task-optimization such that 'which edge server is the most appropriate to run a specific task'. Some novel works have proposed jointly offloading the computation and caching strategy, using MEC [119]. While MEC is a promising solution to enhance the performance of caching strategies, the computational limitation still poses challenges to achieving it practically. In addition, network slicing and network function virtualization (NFV) are other technologies to relieve the computation constraints and support mission-critical services [120].

2.5.6 Content Encryption

While trying to guarantee user security and privacy in cellular networks, content encryption becomes an undeniable challenge to content caching because they need to be encrypted. For that purpose, most current CDNs use the secured Hypertext Transfer Protocol Secure (HTTPS), which makes content invisible to the MNOs and CDN providers. This complexity worsens when using multiplexed transport protocols like QUIC, higher rate application encryption. In addition, the new Internet privacy services, such as virtual private networks (VPN) and Apple iCloud Private Relay, make visibility and management by service providers more difficult. They cannot consent to the property of the content and manage the caching. Thus, it is hard to deal with encrypted content via the traditional caching mechanism, so examining the trades-off privacy of the users and the QoE became very interesting.

Chapter 3

Dynamic Programming-based Caching in Cellular Networks

This chapter explains the developed content caching strategy to serve content requests effectively. It begins with an introductory background and presents research motivation in Section 3.2. Section 3.3 explains a simplified system model formulation, while Section 3.4 details the proposed DP-based strategy. Section 3.5 explains the performance of the DP-ZoSKP strategy using extensive system-level simulations.

3.1 Background

Due to the fast development of the wireless infrastructure and having an elastically swelling number of smart devices, its multimedia content delivery is increasing yearly. In addition, the emergence of new technologies such as the IoT, M2M, e-health, and vehicular communications has pushed cellular traffic much higher than anticipated, with the explosive growth of video traffic [49]. A large portion of the mobile data traffic is video content, which accounts for 69% of the total traffic and is predicted to exceed 79% by 2027 [2]. Such a tsunami of video traffic has instantiated a paradigm shift in the design and implementation of mobile networks, bringing edge content caching capabilities to the research community's center of attention. Notably, the network data traffic is dominated by a few popular videos (compared to the full list of available contents) that users frequently request.

A critical mass of content requests goes to popular ones, often consumed by different end users spanning many mobile helpers and wider geographical regions. Such requests place a great burden on the backhaul of cellular networks by: i) creating redundant re-transmissions of the same content from a central server to the network edge, which creates service cost, ii) using the bandwidth-limited backhaul, and iii) at the peak time, so that causes congestion and transmission failure. As a relief, with the integration of the MEC capabilities in the 5G mobile cellular networks, CDN operators can place popular contents closer to the network edge at an off-peak time. Accordingly, caching these contents into the network edge brings them closer to the end users, avoiding fetching the same content multiple times from the backhaul network.

The content placement strategy is usually formulated as a constrained problem, modeled using different mathematical frameworks. Depending on the proposed placement model, different algorithms are used to solve it. The survey work in [32] reviews the most widely used modeling methods, including game-theoretic, stochastic, and predictive approaches. The authors in [34] use Multiple-choice Knapsack Problems to model a cooperative placement problem in large instances. The emphasis was on caching layered videos using a fully polynomial time approximation (FPTA) algorithm. The authors in [30] model content placement as a reward maximization problem and solve it by reducing it to solvable linear programs. Similarly, the authors in [36] use mixed integer programming to model the cooperative content placement and employ greedy algorithms to solve it. In [12], the authors use linear integer programming to model the content placement problem and apply Lagrangian relaxation with hierarchical prime-dual decomposition to decouple its structure into two levels, enabling solutions using the sub-gradient method. However, calculating optimal multipliers for the Lagrangian relaxation and the sub-gradient methods in large-scale networks is computationally intensive. Other interesting works, such as [46] and [34], model the caching process by the Knapsack problem. In contrast, authors in [12] use integer linear programming with an end-to-end approach which is impractical to solve optimally. Authors in [121] use the Markov decision process and solve it by dynamic programming for cloud resource management.

3.2 Research Motivation

So far, many caching strategies have been proposed to boost the effectiveness of mobile data delivery in cellular networks. However, most of these strategies focus on heuristic methods of traffic characterization, which are inaccurate in representing real networks. Also, to solve them analytically, most research works use greedy methods, which fail to tackle practical constraints of the content placement, *e.g.*, varied content size and popularity, and limited cache size. Though we do not rule out these approaches, as they are computationally lighter, they do not give an optimal solution. Some other strategies rely on information processing techniques, *e.g.*, interference cancellation, and modulation, to improve physical layer performance [90] but do not substitute caching.

The content caching, whose *placement* and *delivery* phases are interdependent, is widely believed to be a dual-nested problem. Thus several types of mutually placing and delivering optimization schemes are proposed[10]. However, we stick to the fact that we can achieve efficient content caching mainly by optimizing the placement phase for three reasons: i) the placement is done at an off-peak time and may wait for multiple epochs, while delivery is an instant action [122], ii) due to extremely dynamic behavior of UEs, the placement is done almost without knowing current statistics of the UEs, iii) the placement is content-centric while delivery is user-centric , *i.e.*, as we focus on caching contents at MHs, making MEC decisions based on its aggregated RAN data analytics is more effective than using individual UE data.

Rather than exhausting the intractable end-to-end optimization approach, we resorted to the dual-nested formulation to solve the content placement phase by maneuvering an efficient modeling tool. We can then apply an optimal algorithm to achieve the required QoE. The complement intuition is that if we work on placing contents at their closest local MH (, *i.e.*, the UE associated to), it is expected to have the lowest service cost and maximum request probability. This low cost is because, without loss of generality, the cost is a function of key parameters such as energy consumption and delay, which depend on the distance between an MH and requesting UE.

We need to apply an effective model that accurately represents content caching from a single source to a single destination with diversified attributes and scarce resources. We typically modeled the content placement problem by the 0/1-Knapsack Problems method. In addition, we aim to achieve an optimal solution to the model to meet the required QoE. Then, we optimally solved the problem with the powerful DP algorithm using the tabular approach. The performance of the proposed novel placement strategy is evaluated, in terms of a cache hit probability at the network edge, by comparing it with the widely used baseline strategies.

3.3 System Model Design

We consider the downlink direction of a heterogeneous cellular network, as shown in Fig. 3.1. The network has several macro base stations (MBS), acting as a source of content to small base stations within their coverage area (*i.e.*, two-tier content placement hierarchy). While making cache placement, the SBSs act as mobile helpers, which is a relay in the network. The MBS and MHs are the first and second network tiers, respectively. Each user equipment is assumed to be associated with a nearby SBS (*i.e.*, the MH). The UE requests contents independently from the appropriate cache-enabled MH, where contents are cached based on popularity.

An MBS is assumed to have a complete list of popular contents that we denote by $\mathcal{M}$, $\mathcal{M}$=$\{(f_m, s_m) : 1 \leq m \leq |\mathcal{M}|\}$, where f_m is the unique identifier (*e.g.*, URL) and s_m is the size of the content f_m, measured in bits. The size of these contents is not necessarily equal. We further assume that each MH has its known local popularity vector (LPV), denoted as $\mathcal{P}$=$\{\rho_1, \rho_2, \cdots, \rho_{|\mathcal{M}|}\}$. Here, content popularity (ρ_m) refers to the probability that an associated UE requests the respective content (*i.e.*, different areas may have different popularity vectors). Without loss of generality, we assume that the LPV is derived through localized estimation techniques employed by each MH, *e.g.*, based on the frequency of requests per content.

In this chapter, we focus on the caching process taking place at tagged MH with its LPV. The LPV is normalized to the number of contents of in $\mathcal{M}$, *i.e.*, $\sum_{m=1}^{|\mathcal{M}|} \rho_m$=1. We assume that content placing processes at one MH is independent of any other MH. That means there is centrally controlled caching in the MHs. Let $\mathcal{C}$ denote the set of contents eventually cached at the tagged MH ($\mathcal{C} \subseteq \mathcal{M}$). The logical representation of content caching is shown in Fig. 3.2.

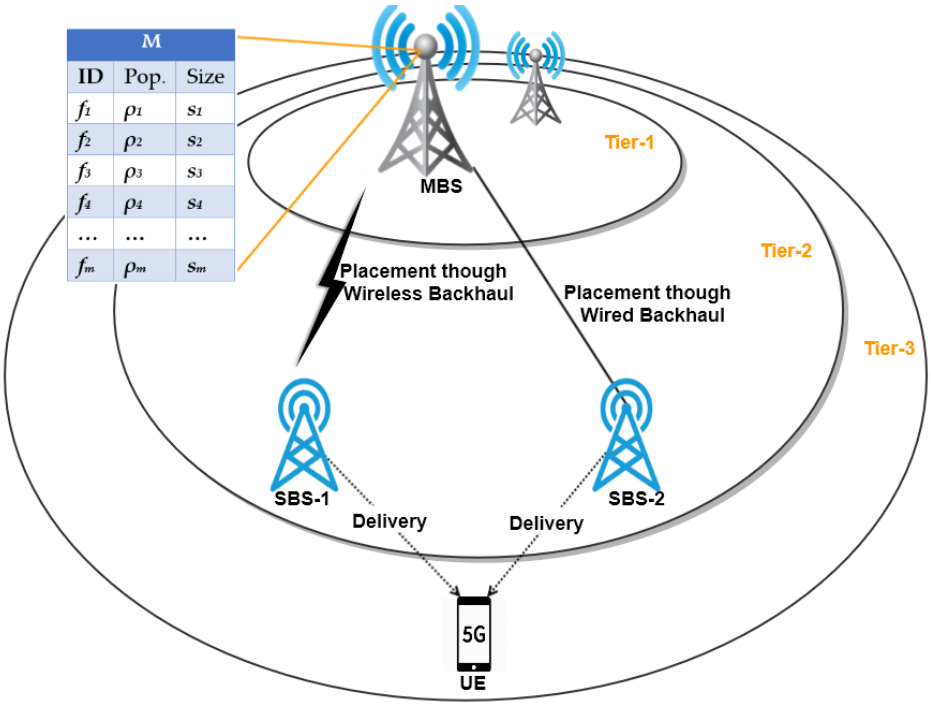

FIGURE 3.1: Content caching system model.

Our objective function is *cache hit probability* of eventually cached contents at the MH. The CHP refers to the probability that content requested by a UE is found cached and can be retrieved from the located MH. Given the available $\mathcal{P}$ and list of cached contents $\mathcal{C}$ at the MH, the CHP is given as follows:

$$\Psi_{\mathcal{M}}(\mathcal{P},\mathcal{C}) = \sum_{f_m \in \mathcal{C}} \rho_m \tag{3.1}$$

Our goal is to select an optimal list of contents from the main library $\mathcal{M}$ and place it in $\mathcal{C}$ that maximizes the CHP at the tagged MH with minimized computational time. While doing so, we assume the contents cannot be partitioned, and only one copy of each is cached. However, the sum of all contents should not exceed the cache size of the MH. We can define the combinatorial problem in terms of maximizing the objective function CHP as follows:

$$P_{1.1}: \max_{C \in 2^{|\mathcal{M}|}} \Psi_{\mathcal{M}}(\mathcal{P},\mathcal{C}) \tag{3.2a}$$

$$\text{subject to:} \quad \sum_{m=1}^{|\mathcal{M}|} x_m \cdot s_m \leq L \tag{3.2b}$$

$$x_{m,n} \in \{0,1\}, 1 \leq m \leq |\mathcal{M}| \tag{3.2c}$$

$$\sum_{m=1}^{|\mathcal{M}|} \rho_m \leq 1 \tag{3.2d}$$

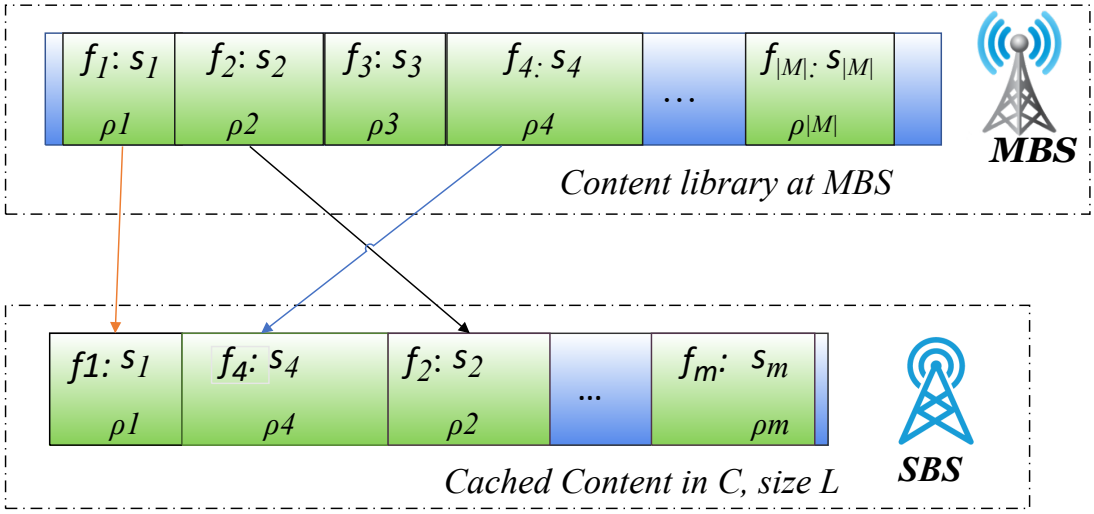

FIGURE 3.2: A caching scheme that selects contents to maximize the CHP.

where the parameter x_m is a cache decision indicator that shows whether f_m ($f_m \in \mathcal{M}$ is selected to be cached. That means, x_m=1 if the event $f_m \in \mathcal{C}$ holds true and x_m=0, otherwise. The optimization problem in Eq. 3.2 is mapped to the well-known 0/1-Knapsack Problem with a few constraints. In Eq. 3.2a, the term $2^{|\mathcal{M}|}$ denotes the power set of content combinations in $\mathcal{M}$, within which an optimal solution is found. The constraint (3.2b) shows that the total size of cached contents per MH should not violate its cache size, while constraint (3.2c) ensures that contents are not partitioned during caching. The last constraint (3.2d) indicates that the LPV of contents in the library is normalized to one, as seen with respect to each MH.

3.4 DP-based Caching Strategy

Recall that while caching, we do not partition contents into chunks, *i.e.*, contents are considered indivisible objects that are either cached as a whole or not cached at all. Instead, using their size and popularity, we enumerate all possible combinations of contents that can fit the cache size. Among the combinations, we take the one that gives the maximum CHP value. The problem $\mathbf{P}_{1.1}$, modeled in formulation Eq. 3.2, can be optimally solved using brute force (exhaustive search of the optimal CHP) on the power set of $\mathcal{M}$. But this approach has $O(2^{|\mathcal{M}|})$ time complexity, which is not computationally achievable and also not scalable as $|\mathcal{M}|$ increases. Instead, the problem is solved using dynamic programming by breaking the original problem into subproblems and solving them sequentially. In this approach, the solution of every subproblem is memorized and reused to solve other subproblems, while the global solution is found by combining these multiple sub-optimal solutions.

Algorithm 1: Dynamic programming bottom-up computation

1 **Input:** $\mathcal{M}, \mathcal{P}, \mathcal{S}, L$
2 **Output:** $\Psi^*_{\mathcal{C}_n}, \mathbf{x}$
3 **Function:** `DP-ZoSKP`$(\mathcal{M}, \mathcal{P}, \mathcal{S}, L)$
4 $\quad \mathbf{V} = Zeros(|\mathcal{M}|, L)$;
5 $\quad \mathbf{x} = Zeros(1, |\mathcal{M}|)$;
6 $\quad$ **for** $1 \leq m \leq |\mathcal{M}|$ **do**
7 $\quad\quad$ **for** $0 \leq j \leq L$ **do**
8 $\quad\quad\quad$ **if** $j \geq s_m$ **then**
9 $\quad\quad\quad\quad V[m, j] = max(V[m-1, j], \rho_m + V[m-1, j-s_m])$;
10 $\quad\quad\quad$ **else**
11 $\quad\quad\quad\quad V[m, j] = V[m-1, j]$;
12 $\quad\quad\quad$ **end**
13 $\quad\quad$ **end**
14 $\quad$ **end**
15 $\quad \Psi^* = V[|\mathcal{M}|, L]$;
16 $\quad m = |\mathcal{M}| + 1, j = L + 1$;
17 $\quad temp = V[m, j]$;
18 $\quad$ **while** $m > 0$ && $j > 0$ **do**
19 $\quad\quad$ **if** $temp \neq V[m-1, j]$ **then**
20 $\quad\quad\quad x_m = 1, \mathbf{x} \leftarrow x_m$;
21 $\quad\quad\quad j = j - s_m$;
22 $\quad\quad\quad m = m - 1$;
23 $\quad\quad\quad temp = V[m, j]$;
24 $\quad\quad$ **else**
25 $\quad\quad\quad m = m - 1$;
26 $\quad\quad$ **end**
27 $\quad$ **end**
28 $\quad$ **return** Ψ^*, $\mathbf{x}$
29 **end**

Though there are various ways of applying the DP, we chose the memoization method for its simplicity in tracing back. The DP pseudocode shown in Algorithm 1 is a tabular memoization method that iteratively examines different content combinations versus the cache size. It begins by initializing the CHP table $V[m, j]$, for m=0, which means no item is selected (lines 4-5). In addition, vector of decision indicators $\mathbf{x}$ is initialized, $\mathbf{x}$=$\{x_m : 1 \leq m \leq |\mathcal{M}|\}$. Lines 6-14 constitute a loop to sequentially calculate all contents, starting from one to the entire set in $\mathcal{M}$. It saves the corresponding CHP value in the array. The algorithm increments the objective value only if the examined content brings a better CHP value (line 11). Then, the optimal CHP value ($\Psi^*_{\mathcal{M}}(\mathcal{P}, \mathcal{C})$) is the last entry of the table, *i.e.*, $V[|\mathcal{M}|, L]$ (line 15). In lines 16-17, we initialize the necessary parameters to trace the contents that lead to the optimal CHP value. Meanwhile, the optimal set of contents is traced back to points where the content examination had

brought a change and registered them (lines 18-27). In the end, the bottom-up computation returns the set of contents with its optimal CHP value.

The caching process using the DP method takes a pseudo-polynomial time of $O(|\mathcal{M}| \cdot L)$) complexity, which is very low compared to the optimal solution called brute-force time complexity. This approach gives an optimal solution at trading off computation cost, mainly the memory for processing large library and cache size. Because the strategy is DP-based and modeled by the 0/1-Knapsack problem, we call this proposed content caching a 'zero-and-one knapsack problem' *DP-ZoSKP* strategy.

3.5 Results and Discussion

In this section, we evaluate the performance of the proposed content placement strategy (*i.e.*, the DP-ZoSKP) by comparing it with two baselines: *Greedy* and *Random*, which are very common in state-of-the-art. The Greedy strategy orders the contents in descending of their popularity and caches the most popular first to the MH until no other (complete) content can fit into the cache. The Random strategy selects contents arbitrarily and caches them in the MH until the storage is fully utilized.

Unless otherwise stated, for this simulation, we consider the number of popular contents in $\mathcal{M}$ to be $|\mathcal{M}|$=100. Depending on the scenario under the scope, we investigate the performance of the content placement strategies under two different popularity distributions which are widely used in the literature: the *uniform* and the *Zipf* [46]. Uniform popularity assumes contents of equal popularity (*i.e.*, $\rho_m=\frac{1}{|\mathcal{M}|}$ for every $m \in \mathcal{M}$). In contrast, Zipf popularity assumes that the popularity of the m^{th} ranked content f_m is expressed as $\rho_m=\frac{m^{-\gamma}}{\sum_{j=1}^{|\mathcal{M}|} j^{-\gamma}}$, where $\gamma \geq 0$ is the Zipf parameter that indicates the popularity skewness. Zipf popularity is widely used as the reference model for YouTube videos [33]. Note that as the value of γ increases, a smaller amount of videos in $\mathcal{M}$ exhibit high popularity, and thus, the popularity is concentrated in a few videos. For example, if the popularity skewness parameter γ=0, the contents have uniform popularity in $\mathcal{M}$. On the other hand, as the value of γ decreases, the popularity is distributed in more videos, and thus, a larger number of videos of lower popularity are encountered. For this work, we chose the Zipf parameter γ=1, which means that the popularity is highly concentrated in fewer number contents, and since $\gamma \geq 1$, the network will have a higher successful transmission probability [17], [24].

One step further, the size of popular contents in $\mathcal{M}$ is adapted according to the Google bitrate recommendation for Standard Dynamic Range (SDR) video upload in YouTube [123], and the NTT-DOCOMO guideline for video delivery in mobile networks [124]. In particular, depending on the scenario under simulation, we consider two different content size values: 1.23 Gb, which corresponds to a video duration of 4 minutes with 720p resolution at a 5 Mbps rate, or 8.43 Gb, which corresponds to a video duration of 4 minutes with 2160p resolution at 35 Mbps rate. Note that the duration has been selected in line with the study in [125], which reported that video duration of 1 and 4 minutes are among the most common in video consumption. Subject to changes for some scenarios, we further consider that the s_m is either fixed or

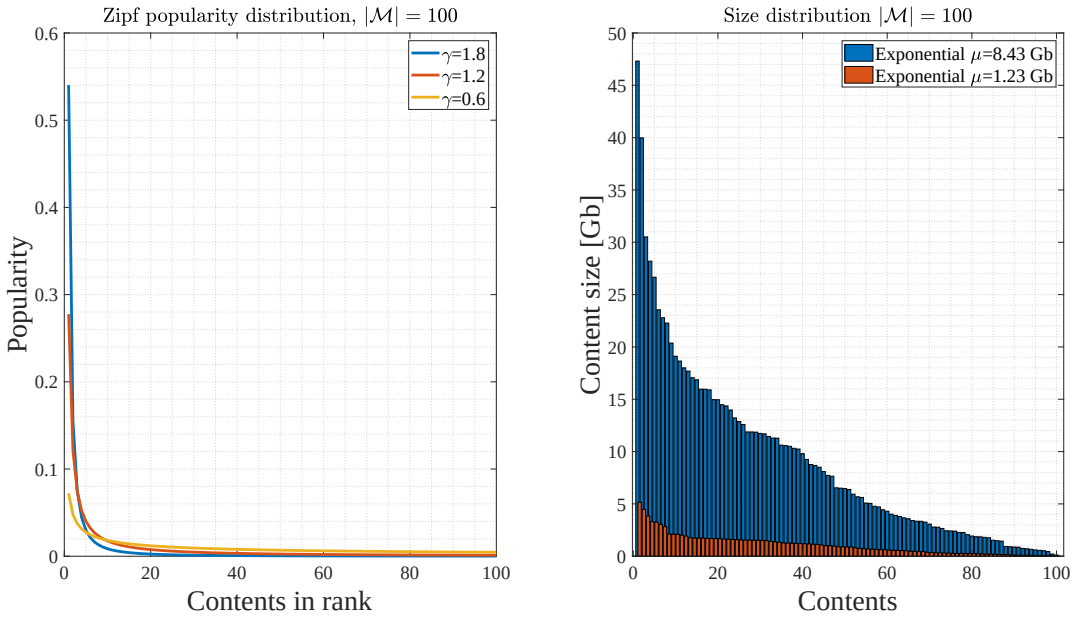

FIGURE 3.3: Popularity distribution (left) and Exponential content size.

exponentially distributed with a mean value of 1.23 or 8.42 Gb. The characteristics of the content popularity Zipf distribution and content size exponential distribution, as a reference for all related sections in this dissertation, are shown in plots in Fig. 3.3.

In the following subsections, we investigate the performance of the content placement strategies assuming: i) uniform content popularity with exponentially distributed size (Section 3.5.1), ii) Zipf content popularity with equal content size (fixed) (Section 3.5.2) and, iii) Zipf content popularity with exponentially distributed content size (Section 3.5.3). For the first two scenarios, our analysis aims to reveal the conditions under which the Greedy content placement is sufficient to achieve optimal performance. The third scenario assesses the performance gains attained under more realistic network deployments (Zipf popularity and varying content size).

3.5.1 Uniform popularity & exponential size distribution

In Fig. 3.4, we evaluate the performance of the three content placement strategies under the scope, assuming that: i) all contents have *uniform popularity* and, ii) their size is exponentially distributed with mean values (μ)=1.23 Gb (4 minutes video at 720p with 5 Mbps rate), or μ=8.43 Gb (4 minutes video at 2160p with 35 Mbps rate). Since all contents have equal popularity, in this analysis, we differentiate the performance of the Greedy strategy depending on the logic used to prioritize contents of different sizes. For that, we used three terms: i) *Greedy-ascending-size*, the greedy strategy where videos of lower content size are given higher priority of selection; ii) *Greedy-descending-size*, the greedy strategy where videos of higher content size are given higher priority and; iii) *Greedy-random-size*, the greedy strategy where the content size is not taken into account when caching popular contents at the MH. First, let us focus on the group

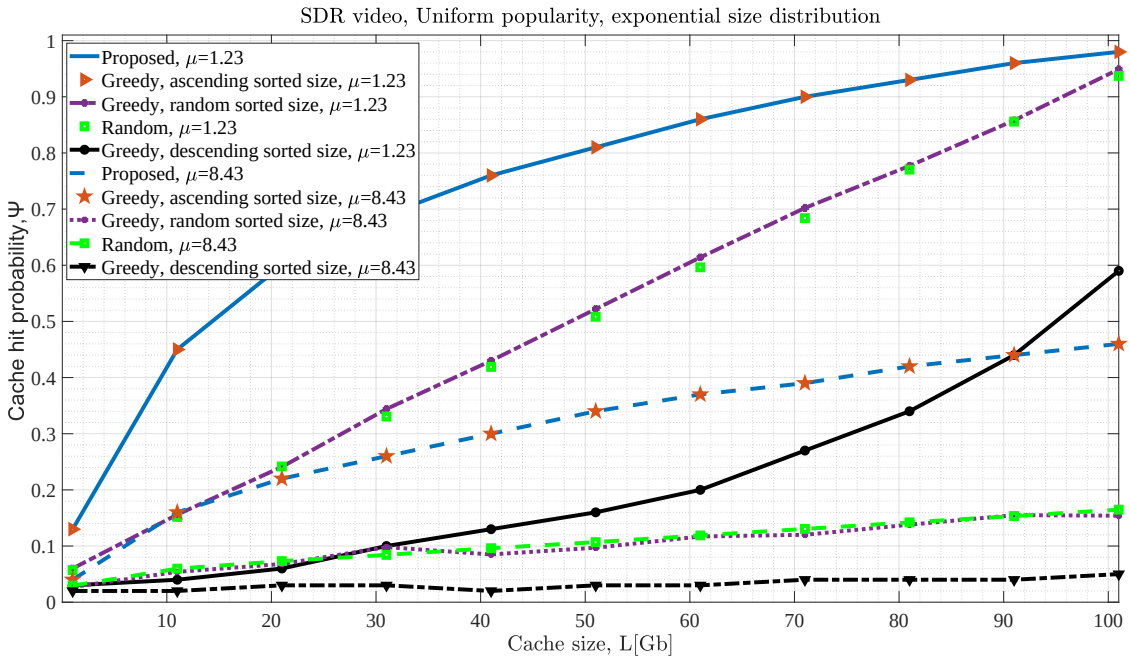

FIGURE 3.4: CHP for uniform popularity and exponential content size.

of results for the mean content size μ=1.230 Gb. As expected, the proposed content placement strategy attains optimal performance, improving quickly as the cache size increases. The performance of the Greedy-ascending-size strategy, which prioritizes content of lower content size, matches the proposed strategy (optimal performance). This readily follows from the fact that $\mathcal{M}$ includes videos of equal content sizes, so the prioritization of contents with smaller sizes gives optimal performance as well (by the principle of bin packing problem).

The plots in Fig. 3.4 also show that the proposed and the Greedy-ascending-size strategies outperform the Random and the Greedy-random-size strategies, attaining even a triple-fold increase in the cache hit probability (*e.g.*, for L=10 Gb). The two random strategies on size (Random and Greedy-random-size) perform equally because they do not treat the size constraints. However, the performance gap between the optimal (proposed and Greedy-ascending-size) and the Random strategies decrease fast as the available cache size becomes comparable with the content library size (close to the value of $|M| \cdot \mu$). On the other hand, prioritizing large content sizes with equal popularity (Greedy-descending-size) is shown to attain very low performance compared to other strategies because the cache gets fully utilized with only a few contents.

Second, we focus on the case μ=8.43 Gb results. Similar performance trends are shown for this case in all content placement strategies under the scope. However, it further tells us that when there are large contents, the cache storage is exhausted with a few numbers of contents. Interestingly, by comparing the results of the proposed strategy for μ=1.23 Gb and μ=8.43 Gb, we observe that a multi-fold increase in the mean content size does not decrease the CHP proportionally with the same rate (*i.e.*, the CHP does not decrease with the rate of $\frac{8.43}{1.23}$=6.85).

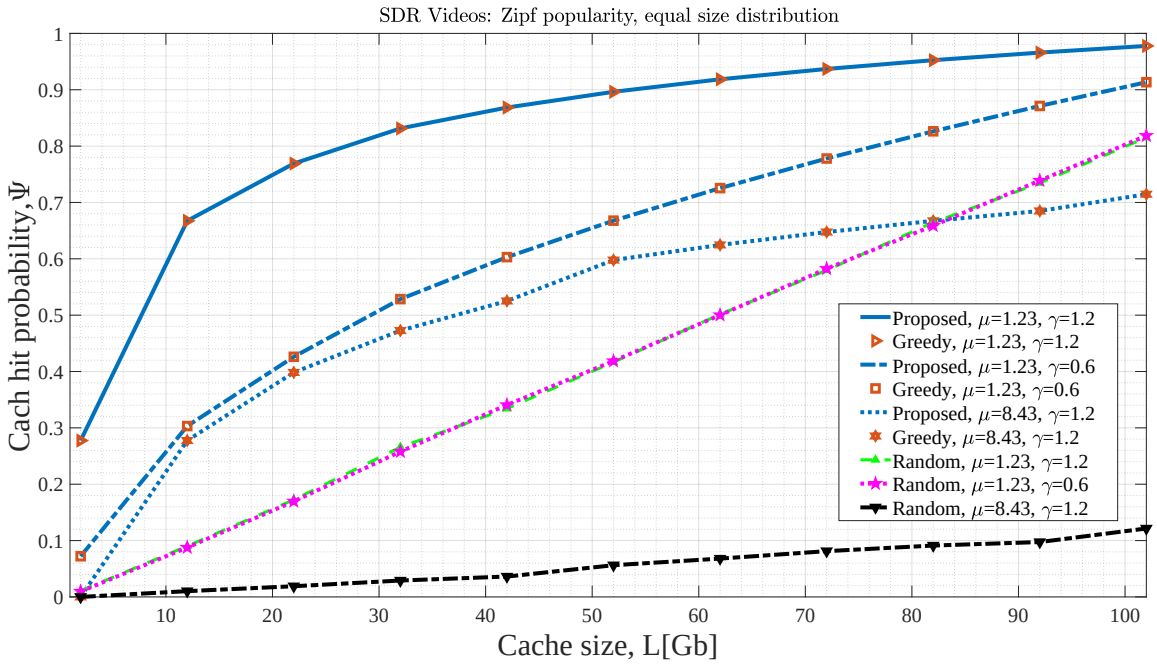

FIGURE 3.5: CHP for Zipf popularity and equal content size.

The main contribution of this scenario is, of the two cases, that if the content popularity is uniform, but we cache shorter files first, optimal caching is found, which equally performs with the proposed DP-based strategy.

3.5.2 Zipf popularity & equal content size

In this scenario, we assess the performance of the content placement strategies assuming Zipf popularity distribution and equal content size. Note that an increase in the value of the Zipf parameter γ (higher skewness) corresponds to a decrease in the number of contents that account for the same aggregate popularity in the library. For example, in the case, γ=0.6, only 10% of the contents account for 32.12% of the popularity in $\mathcal{M}$. When we increase the skewness value, say γ=1.2, only 10% of the library contents account for a higher value of 68.5% popularity in $\mathcal{M}$. The simulation results of this scenario are plotted in Fig. 3.5, in descending order of performance. The plots, three per strategy, indicate the impact of the γ and content size s_m.

The proposed DP-ZoSKP content placement strategy leads the performance under all scenarios within the scope. The greedy strategy, which prioritizes contents with higher popularity, equally outperforms the proposed strategy, given that all contents have equal sizes. This result is, in effect, independent of the γ value because both strategies attract the most popular ones, but the size is not a contesting parameter among contents. In contrast to the random strategy, the performance gains attained by the proposed and the greedy strategies are very high, especially when the cache size

is small and comparable to the size of popular contents (fixed to 1.23 Gb- 4 minutes, 720p, 5 Mbps and 8.43 Gb- 4 minutes, 2160p, 35 Mbps videos).

Going deeper by doubling γ from 0.6 to 1.2, for s_m=1.23 Gb, gives up to a doubling CHP value, subjected to cache size. But, interestingly, the performance of the random strategy is not affected by the γ parameter (*i.e.*, similar performance for γ=0.6 and γ=1.2) because the performance of a random selection of cached contents remains unaffected by the skewness.

Once again, increased content size is shown to reduce the CHP performance; but for this case, as compared to the results of the proposed strategy in Fig. 3.4, the performance gap between cases μ=1.23 and μ=8.43 is lower due to the nature of the Zipf (instead of uniform) popularity distribution. This gap gets lower when the cache size is larger, meaning more contents have a chance to be cached. In all use cases, an increase to the Zipf parameter greatly improves the performance of the proposed and the greedy strategies, indicating that a higher skewness in the distribution of popular contents is highly beneficial for the CHP performance provided the employed content placement strategy can leverage this feature (*e.g.*, random strategies can not).

3.5.3 Zipf popularity & exponential size distribution

In this scenario, we focus on the most realistic cases of the HCN where the popularity of video library $\mathcal{M}$ is characterized by the Zipf distribution, and content size is modeled using the Exponential distribution of mean μ=1.23 Gb (4-minute video of 720p at 5 Mbps rate). We evaluate the performance of the three main content placement strategies under three different γ values. Note that γ=1.8 corresponds to the scenario where roughly 10% of the library contents account for 91.4% of the popularity in $\mathcal{M}$.

As observed in Fig. 3.6, the proposed optimal content placement strategy outperforms all competing strategies. However, the performance gap between the proposed and the greedy strategies is very small for skewed popularity (such as γ=1.8) but increases when the skewness decreases (*i.e.*, for γ=1.2, the proposed outperforms the greedy strategy by an average of 50%). This trend follows from the fact that as a smaller number of contents in the video library $\mathcal{M}$ dominate the popularity distribution (γ is high), selecting only a few of these files gives high CHP, which also achievable by the greedy prioritization. However, when the popularity distribution exhibits a lower skewness, the performance gap between the greedy and the proposed strategy increases rapidly (for a proportional reduction of the Zipf parameter γ), reaching up to 118% for γ=0.6. Similar to the results of Fig. 3.5, the CHP of the random strategy increases linearly with the cache size but remains unaffected by an increase of the γ.

3.5.4 Cache buffer utilization

Before caching buffer utilization as a resource, let us take a deeper insight into content size diversity by comparing Fig. 3.5 (where s_m=1.23 Gb) and 3.7 (where μ=1.23 Gb), at γ=1.2. The same parameters, in the second case, show that content size is diversified, where the proposed strategy outperforms the greedy one by an average of 104%

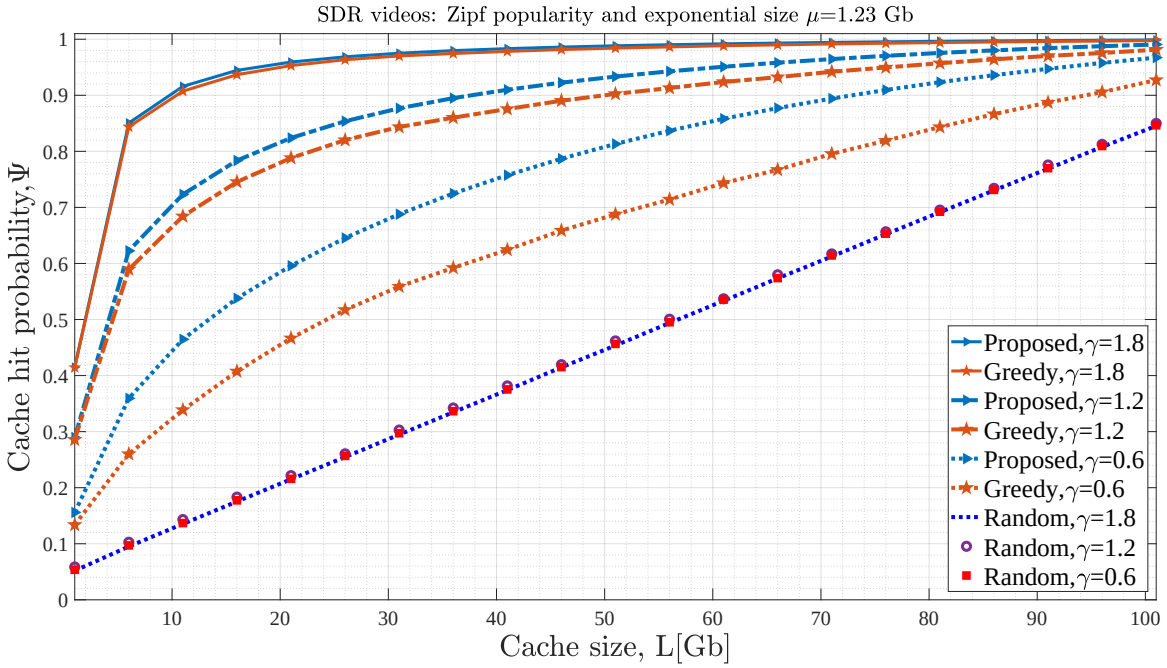

FIGURE 3.6: CHP for Zipf popularity and exponential content size.

improvement. This outcome implies that the proposed strategy is much preferable for the case of higher content diversity because it selects contents that bring better CHP for available cache space.

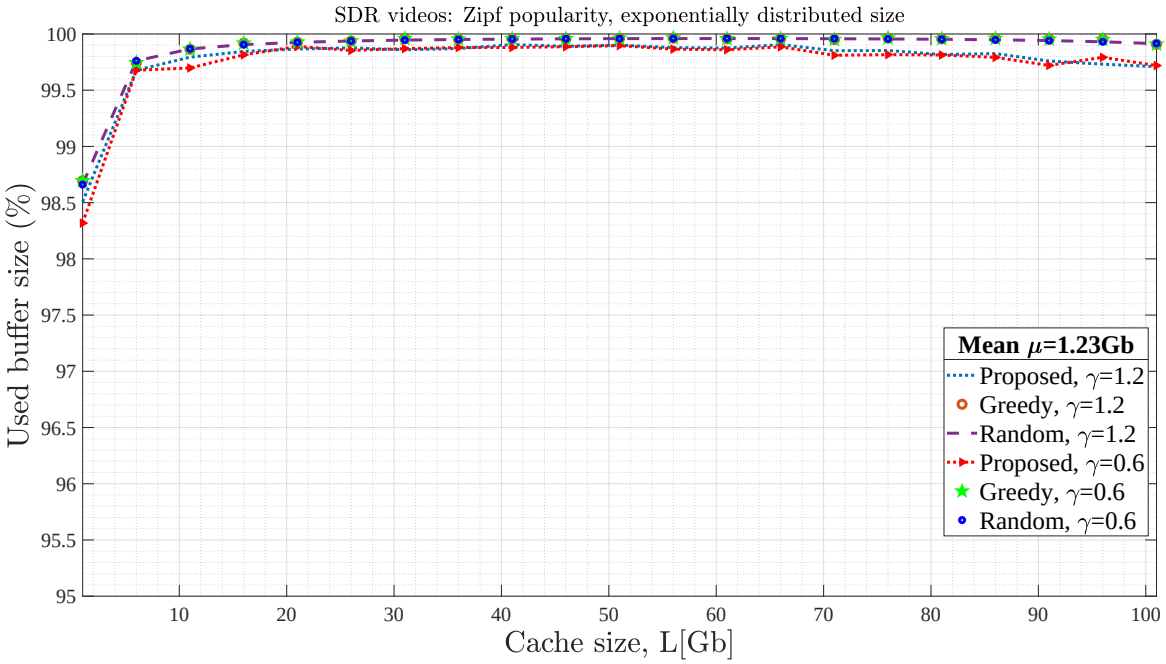

FIGURE 3.7: Used buffer for Zipf popularity and exponential content size.

As shown in Fig. 3.7, the proposed strategy outperforms at a lower expense of cache buffer, regardless of the γ value, which gives effective resource utilization. This performance is because the proposed strategy fills the cache space more effectively than the baseline strategies. However, the residual buffer has no relation to the cache hit probability. The two baseline strategies have nearly equal buffer size usage.

3.6 Summary

In this chapter, we have studied optimal content placement strategies in cellular networks where small cells act as relays by caching contents to offload macro cell base stations. We modeled the popular content caching strategy within a limited cache size resource using the 0/1-Knapsack problem. We then used dynamic programming to solve it optimally. The performance of the obtained optimal solution is evaluated and compared with other baseline placement strategies. Useful guidelines for the design of cache-enabled cellular networks are also given.

The main contribution of this chapter is we have delivered an optimal and enumerative caching strategy. Very useful design guidelines are given where the proposed strategy is recommended, such as for real networks, γ close to 1, and μ=1.23 Gb. The impact of critical system parameters on the CHP performance has also been thoroughly investigated in practical network deployment scenarios where the content popularity follows the Zipf distribution, and the size is exponentially distributed. Both content popularity and size diversities, as part of the HCN heterogeneous property, are effectively managed by the proposed strategy. This contribution is a building block for the extension works, explained in the next consecutive chapters. Future work includes deploying the strategy using collaboration across the MHs and other evaluation metrics such as success probability.

Chapter 4

Cooperative Caching to Multiple MEC-enabled Edges

In this chapter, we describe the proposed content caching strategy in multi-tier HCN where several caching edges cooperate in a cluster to serve correlated video requests. The introductory note in Section 4.1 summarizes related works of the topic, highlighting modeling and optimization approaches. Section 4.3 introduces the system description and the problem formulation for cluster-based caching. Similarly, Section 4.5 presents the proposed bound-and-bound ZoMKP (BB-ZoMKP) placement strategy in a block-by-block fashion. In the end, Section 4.6 briefs the performance evaluation of the proposed strategy using extensive system-level comparative simulations.

4.1 Introduction

The HCN is often upgraded to accommodate the offered mobile data traffic load, *e.g.*, by deploying new generation backhaul and fronthaul technologies. However, the exponential increase of mobile content traffic still brings the utilization of the existing mobile backhaul links to their capacity limit. This limit raises questions about the scalability of existing systems to meet the ever-increasing demand for seamless content delivery. Mobile network traffic is also overwhelmed by multiple requests for some popular content, creating a redundant end-to-end content delivery chain from the requesting UEs to the CS. Thus, contents are repeatedly re-transmitted from the CS on the far Internet, causing unnecessary utilization of intermediate network resources. This problem is further exacerbated in the 5G and beyond mobile networks, where low latency requirements of less than 1 ms and a very high level of reliability of 99.99% are targeted for specific service types that co-utilize the wireless medium [126]. So content caching becomes a robust method to meet the high-end performance targets set for future mobile networks of higher deployment density as a cost-effective capacity-boosting network-layer solution [127], [128].

Besides the caching technology, MEC integration into RAN overplays the caching performance by enabling the MNOs to cope with content popularity dynamics over time and across geographical regions. The MEC executes cache computations in the actual HCN topology and improves its performance, especially for the seamless delivery of personalized video content. In doing so, MEC-enabled MHs can [129]: i) predict

content popularity to store videos locally in the user area and offload the RAN during peak periods (reducing user-perceived latency), ii) employ local transcoding on high-resolution videos to match the screen resolution of mobile terminals (enhancing user-perceived rate) and, iii) move relevant content and context closer to the end user (increasing the user-perceived content availability) [3]. In the sense of cooperation, if a requested content is found in the cluster, all corresponding MHs share resources with the *local* MH (*i.e.*, the edge where the original request made by the UE) [11]. Accordingly, requested contents are relayed from a caching MH to the *serving* MH (the edge that finally delivers the content to the user).

In [46], authors describe a cooperative multi-tier caching system decomposed into a series of independent Knapsack subproblems, which are solved using greedy methods. Focusing on adaptive streaming, the authors in [130] use a polynomial time greedy method to solve a series of knapsack problems to cache different video versions at the network edge. Authors in [33] model the content placement problem using a stochastic approach and have applied a fully polynomial time approximation method where a random set of contents is cached to the network edge, placement probabilities within a tier are considered to be the same. In [14], the authors employ stochastic geometry under fixed cache size and bandwidth constraints, assuming a greedy method to solve the proposed problem. In [38], authors apply the Lyapunov function to allow hybrid cloud and edge content caching using greedy and heuristic placement algorithms. In [35], authors model the placement using the Multiple Knapsack Problem. They apply the DP to solve the uncoded caching case, but this profit maximization problem still depends on scaling approximations before the DP is applied.

4.2 Why Need Another Strategy?

The widely used greedy and heuristic approximation algorithms could be computationally feasible for large-scale networks but exhibit sub-optimal performance. This gap shows that extra effort is required to deal with the inhomogeneous content popularity, content size, and caching storage over large geographical areas. In addition, we need to deal with the HCN heterogeneity in sizes and capabilities, *e.g.*, a different number of edges per MEC cluster, and a mixture of MEC edge types.

Unlike many works, we want to manage the *no-partition* and *no-repetition* constraints. Both constraints result in a mathematically complex problem but are of high practical interest in realistic MEC-enabled setups for some reasons. Firstly, adding and fetching cached segments of a given content from the cache of multiple edges increases the communication and intra-cluster transmission costs and cache monitoring overheads. Also, distributed caching in the MEC cluster requires perfect tracking of cached segments and sophisticated multi-source transmission schemes. Secondly, even though redundant caching of the same content at different MEC edges increases the availability of popular contents within the same cluster, it under-utilizes the cache pool so that the probability of a cache hit events degrades. Accordingly, the two requirements of no-partition and no-repetition are in line with MEC-enabled service provisioning,

which primarily aims to leverage edge network resources and avoids using end-to-end links to the far Internet through the backhaul network.

This chapter investigates how content placement can be formulated and optimized under the emerging MEC integration scenario, where the MNO can pool the network's computation and storage resources to form joint MEC service clusters (areas). Although not included here, the formation of MEC service areas shall enable the MNOs to better adapt: i) the local spatiotemporal user density and request correlation in smaller geographical areas, and ii) the actual information of the RAN topology and capabilities. In such an assumption, the key requirement is that the MEC clustering is performed with insignificant intra-cluster content transmission cost. As an optimization objective function, we use the *cache hit probability* (CHP) at the cluster level, subjected to a set of constraints such as the limited cache sizes, the unique and non-redundant placement of contents per cluster, and without content partitioning.

4.3 System Model Design

We focus on an error-free downlink direction of a three-tier MEC-enabled HCN, where each tier consists of HCN edges of similar networking capabilities. The MNO employs a MEC-cluster formation algorithm to group a heterogeneous set of cache-enabled edges that carries out the edge network caching process.

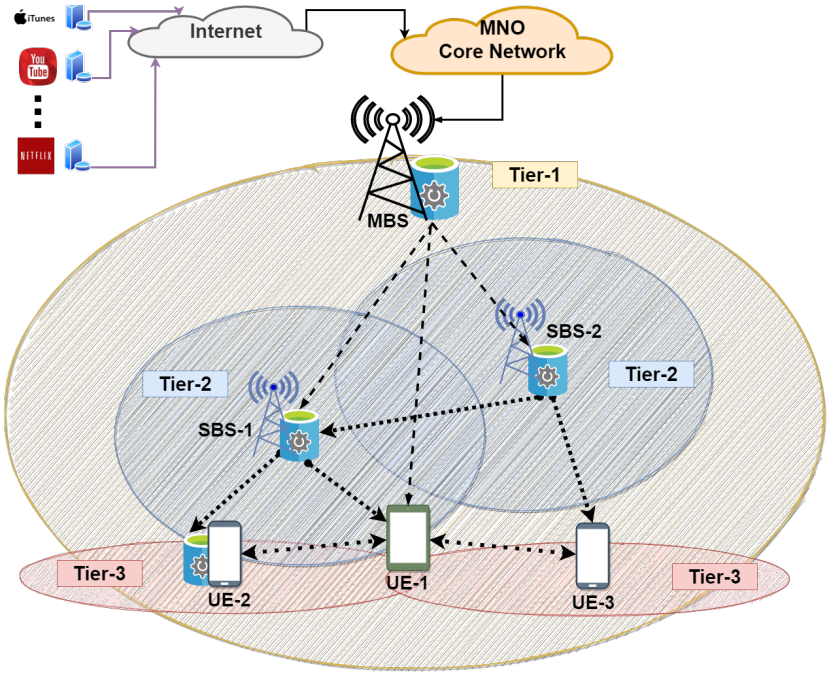

FIGURE 4.1: System model for multiple edge caching.

A cache-enabled MEC cluster can comprise HCN edges belonging to different network tiers. In Fig. 4.1, we provide an illustrative example of a three-tier system model instance where a MEC cluster is formed by MBS, SBS-1, SBS-2, UE-1, UE-2, and UE-3.

Without loss of generality, for each MEC-enabled cluster, we consider that a cluster head (*e.g.*, an MBS) centrally controls the content placement. Cache-enabled edges belonging to the same MEC cluster may have different cache sizes (*i.e.*, storage capacity), where they are considered to follow the instructions of the cluster head to fill their cache with the allocated content during off-peak periods.

We focus on the content placement process for a given period, termed as *time epoch* in a tagged cache-enabled MEC cluster. Let $\mathcal{N}$ denote the set of caching edges and $|\mathcal{N}|$ represent their number in the tagged MEC cluster, and let L_n denote the cache size (in bits) of the n^{th} edge with $n \in \{1, ..., |\mathcal{N}|\}$. And let $\mathcal{M}=\{f_m : 1 \leq m \leq |\mathcal{M}|\}$ denote the set (list of size $|\mathcal{M}|$) of popular video contents, where f_m is the unique identifier of the m^{th} content . Also let ρ_m and s_m denote the popularity and size of a target content $f_m \in \mathcal{M}$, respectively. The popularity is evaluated on a per epoch and cluster basis, while it can match the user request ratio on a per cluster basis for the given popular videos. Accordingly, we define the set of content sizes $\mathcal{S}=\left\{s_1, s_2, ..., s_{|\mathcal{M}|}\right\}$ and the set of popularity values (*i.e.*, the popularity vector) $\mathcal{P}=\left\{\rho_1, \rho_2, ..., \rho_{|\mathcal{M}|}\right\}$, where $\sum_{m=1}^{|\mathcal{M}|} \rho_m$=1. The content popularity values are assumed to be normalized over the set $\mathcal{M}$. In addition, the ρ_m value is estimated by taking the aggregated number of content requests from all UEs through all MHs in a MEC cluster and using Eq. 2.1.

For a tagged epoch, we consider that the MEC cluster head decides on a list of target popular contents, with known popularity and size. The MH deploys its policy to estimate the popularity distribution and infer a list of popular contents. It governs the list within its coverage area, potentially encompassing information from the MNO's core network (*e.g.*, using traffic tracing and popularity estimation techniques [115]). The UEs are associated with a local MH, using the Radio Access Technology (RAT) network discovery and attachment protocols [76]. Then, the local MH first examines whether the requested video is found in its local cache. If a *local cache hit* event happens, it acts as the serving MH and directly transmits the video to requesting UE. If not found, the local MH searches for the requested content within the MEC cluster, *e.g.*, by querying the MH or utilizing an intra-cluster cache map to all MHs.

If the content is found in another MH, called *hosting* MH, a *cluster cache hit* event happens. Then, we must deploy in-cluster content delivery methods to relay the requested f_m to the serving MH. Finally, the serving MH serves content, as shown in Fig. 4.2. Subject to no-partition and no-repetition constraints, in-cluster content delivery becomes effective by implementing direct or multi-hop links between the serving and hosting MHs, depending on the technologies available in the cluster. If the f_m cannot be found in the cache of any cluster edge, then the serving MH shall establish an end-to-end connection to the CS, generally thought to contain the requested content. For example, in Fig. 4.2, the UE requests MH_1 for videos f_1, f_6, and f_{10}. The MH_1 forwards the request for f_{10} to MH_N, which relays f_{10} to MH_1 using in-cluster content delivery. Then, the UE receives f_6 and f_{10} from MH_1 while it gets f_1 directly from MH_2.

At the beginning of each epoch, the MEC cluster head is assumed to be aware of the network information, such as a list of popular contents $\mathcal{M}$, the size of popular contents in $\mathcal{S}$, their popularity values in $\mathcal{P}$, as the set of N, their cache size values L_n.

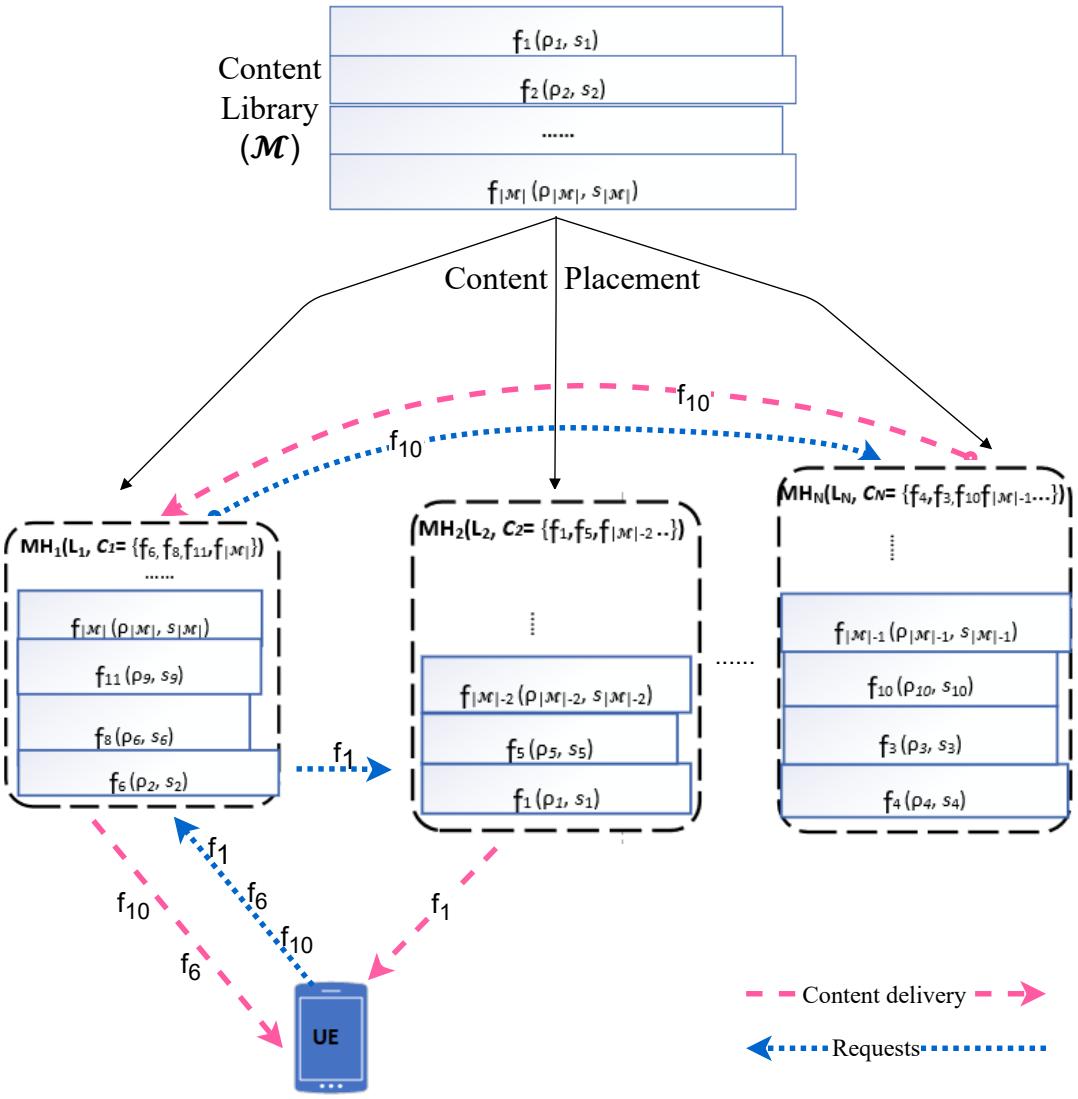

FIGURE 4.2: A logical representation of content caching.

The cluster head implements its content placement algorithm to decide the popular contents that have to be cached in the buffers of all $|\mathcal{N}|$ edges. For a given epoch, the content placement strategy should take into consideration that: i) the number of MHs in the cluster is fixed ($|\mathcal{N}|$), ii) the cache size available per edge is fixed (L_n), iii) the size and popularity of contents in $\mathcal{M}$ are fixed and known (using sets $\mathcal{S}$ and $\mathcal{P}$), iv) cluster edges can relay contents between them only after user requests, and v) $\mathcal{M}$ can contain a much larger set of contents than all requested contents.

After the content placement process, let $\mathcal{C}_n$ denote the set of cached contents at the n^{th} MH for a given epoch. Let also $\mathcal{C}$ represent the full set of cached contents in the MEC cluster. Here, $\mathcal{C}=\bigcup_{n=1}^{|\mathcal{N}|} \mathcal{C}_n$, $\mathcal{C}_n \subseteq \mathcal{M}$, and $\mathcal{C} \subseteq \mathcal{M}$. Given the system model constraints mentioned above, it readily follows that: $\mathcal{C}_i \cap \mathcal{C}_j=\emptyset$, $\forall i,j \in \mathcal{N}, i \neq j$ and $\sum_{f_m \in \mathcal{C}_n} s_m \leq L_n$. In the sequel, the indicator parameter $x_{m,n}$ denotes the event where

the n^{th} MH, in the MEC cluster, has cached content f_m ($f_m \in \mathcal{M}$ and $n \in \mathcal{N}$). That means, $x_{m,n}$=1 if the event $f_m \in \mathcal{C}_n$ holds true and $x_{m,n}$=0, otherwise. All the cluster-wide decision parameters are stored in a lookup table, denoted by $\mathbf{x}$ and defined as: $\mathbf{x}$=$\{x_{m,n} : \forall f_m \in \mathcal{M}, \forall n \in \mathcal{N}\}$. Accordingly, the cluster head requires to maximize the cluster cache hit probability (CHP, described in subsection 2.1.4) of ultimately cached content, a metric that we define based on the popularity distribution on $\mathcal{M}$ as follows:

$$\Psi_{\mathcal{M}}(\mathcal{P}, \mathcal{C}) = \sum_{n=1}^{|\mathcal{N}|} \sum_{m=1}^{|\mathcal{C}_n|} \rho_m x_{m,n} \tag{4.1}$$

4.4 Multiple Knapsack Problem Formulation

Recall that the CHP is defined as the probability that requested video content is found in the cache of any MH belonging to the MEC cluster where the original video request has been made. Provided that content partitioning is not allowed and that MHs belonging to the same cluster cannot cache the same popular video content, it follows that $\sum_{n=1}^{|\mathcal{N}|} x_{m,n} \leq 1$ for $1 \leq m \leq |\mathcal{M}|$. Accordingly, we present the ZoMKP formulation in the context of MEC-enabled cluster-based edge content placement as:

$$P_{4.1} : \max_{x_{m,n} \in \mathbf{x}} \sum_{n=1}^{|\mathcal{N}|} \sum_{m=1}^{|\mathcal{M}|} \rho_m \cdot x_{m,n} \tag{4.2a}$$

$$\text{subject to:} \quad \sum_{m=1}^{|\mathcal{M}|} s_m \cdot x_{m,n} \leq L_n, 1 \leq n \leq |\mathcal{N}| \tag{4.2b}$$

$$\sum_{n=1}^{|\mathcal{N}|} x_{m,n} \leq 1, 1 \leq m \leq |\mathcal{M}| \tag{4.2c}$$

$$x_{m,n} \in \{0,1\}, 1 \leq n \leq |\mathcal{N}|, 1 \leq m \leq |\mathcal{M}| \tag{4.2d}$$

$$\sum_{m=1}^{|\mathcal{M}|} \rho_m \leq 1 \tag{4.2e}$$

In Eq. 4.2a, the $x_{m,n}$ values ($1 \leq n \leq |\mathcal{N}|, 1 \leq m \leq |\mathcal{M}|$) should be adopted by the content placement strategy to maximize the cluster-wide CHP under the constraints (4.2b)-(4.2e). Constraint (4.2b) formalizes that the total size of video contents placed per MH cannot exceed its cache size limit. Constraint (4.2c) limits that popular video contents cannot be cached in more than one MH in the same MEC-cluster (no-repetition) while constraint (4.2d) ensures that content partitioning is not allowed (*i.e.*, no intermediate chunks are enabled). Constraint (4.2e) follows by the construction of the content popularity values summing up to one for all $f_m \in \mathcal{M}$ means that the popularity of content is similar towards all MHs in a cluster.

It is worth noting that if we remove constraint (4.2e), we allow the popularity values to vary towards different MHs and redefined to $\rho_{m,n}$ ($\forall f_m \in \mathcal{M}, \forall n \in \mathcal{N}|$) so that the

edges have different interest for content and not be normalized as probabilities given a distribution over the state space $\mathcal{M}$. This scenario dramatically extends to real network behavior but is extremely complex. On such occasions, the CHP no more works as an objective function while others, such as CHR and utility (reward) functions, should be considered, as discussed in Chapter 5.

4.5 Proposed Combinatorial Optimization

The ZoMKP formulated in problem $\mathbf{P}_{4.1}$ (4.2) is an NP-hard problem that is widely applied in operational research and optimization fields [131], unlike in the cellular network area. In the case of such a problem, solving it optimally is almost surely impossible. Rather, some heuristic and recursive methods are used, which give either sub-optimal or exact solutions. In this section, we introduce an exact *bound-and-bound* (BaB)-based method, originally structured by authors in [132], to the network optimization field by adapting it to the context of our content caching Problem in Eq. 4.2. As a branch, we consider a content placement chain where: i) the placement of a given number of contents has been fixed to the cache of some MHs and ii) multiple options exist for fixing the remaining contents to the cache of some MHs with a non-full buffer.

The proposed BaB-based content placement strategy is a modification of a typical state space exploration technique that progressively fills the caches of MHs, considering both upper and lower performance bounds to avoid evaluating inefficient branches. We consider that the MHs are sorted in increasing order of their cache size, *i.e.*, $L_1 \leq L_2 \leq, ..., \leq L_{|\mathcal{N}|}$. In contrast, the popular videos in $|\mathcal{M}|$ are sorted in decreasing order of their 'popularity per size unit', *i.e.*, $\rho_1/s_1 \geq \rho_2/s_2 \geq .... \geq \rho_{|\mathcal{M}|}/s_{|\mathcal{M}|}$.

The BB-ZoMKP strategy calculates the upper CHP performance bound (the UB) of the original problem, assuming an ideal scenario where contents are placed into a single knapsack of size L=$\sum_{n=1}^{|\mathcal{N}|} L_n$. This size means that the MEC cluster forms a virtual aggregate pool of all MH's cache resources, similar to the surrogate relaxation. The problem above is equivalent to the ZoSKP for which optimal solution algorithms exist using the DP method. On the other hand, a lower performance bound is calculated by iteratively fixing contents into the caches of the MHs. The edges are optimally filled after modeling them with the ZoSKP Problem. Individual formulations are progressively solved (*i.e.*, the MHs are filled one by one, in increasing order of cache sizes) using DP based algorithm, discussed in Chapter 3. In the sequel, popular contents assigned to any MH in previous iterations are excluded from the next iterations.

Based on the LB and UB, the proposed strategy is capable of rejecting infeasible solutions: i) progressively placings contents into the cache of MHs, ii) calculating the corresponding UB and LB performance values conditioned on the above placements, and iii) rejecting solution branches that exhibit poorer CHP performance than the LB, through the use of a backtracking process. A different data stack is used per MH to enable effective backtracking of the partial solution and the exploration of new solution branches (including the re-calculation of upper and lower performance bounds).

In the following subsections, we explain the building blocks of the BaB-based caching strategy to solve the ZoMKP. We progressively present the strategy. The subsection 4.5.1 discusses an optimal content placement strategy for a single MH– the ZoSKP formulation. Subsection 4.5.2 presents the proposed solution for the ZoMKP formulation. Interested readers for more details on why the BB-ZoMKP strategy is exact, if not optimal, are referred to [132] and [133].

4.5.1 Optimal solution for ZoSKP formulation

The content placement to a single MH is modeled by the 0/1-Knapsack Problems and optimally solved using DP [31], [133]. Adapted from the DP-ZoSKP strategy, Algorithm 2 provides the pseudocode of the content placement strategy to solve the ZoSKP for each MH. The algorithm uses as input: a library of free popular videos $\mathcal{M}$, a vector of corresponding popularity $\mathcal{P}_n$ and a vector of content sizes $\mathcal{S}$, and available cache size L_n. We take a two-dimensional matrix V of size $(|\mathcal{M}|+1) \times (L_n+1)$ to memorize different solution branches and infer the optimal solution. Each column-j of V corresponds to a 'unit cache size' of target MH, whereas each row-m indicates the instant evaluation of given f_m. The values of $V[m, j]$ record the highest CHP value that can be attained for the instance where the first m popular contents are to be placed into j cache unit size of MH n. Accordingly, the optimal CHP value is given by $V[|\mathcal{M}|+1, L_n+1]$.

Apart from obtaining the $V[|\mathcal{M}|+1, L_n+1]$ value of each iteration, the algorithm also makes a backtracking process. It traces back and identifies the optimal set of contents, given V, that accounted for the maximum CHP value. This selection indicates which contents have to be cached per MH. Starting from the last entry of V, the backtracking process skips rows (in the current column) that have the same value with $V[|\mathcal{M}|+1, L_n+1]$. When a different value is found: i) the content corresponding to the respective row is included in the solution set, ii) the current row is updated to a new value, and iii) the current column is left-shifted by an equal number of columns with the size of the respective content. The three steps are repeated unless it reaches the first content (f_1) and unit cache size j=1.

The algorithm's output is the optimal content placement for each MH, which is indexed by a global vector $x_{m,n}$ of size $|\mathcal{M}|$. The index vector shows whether f_m is selected for n^{th} target MH such that $x_{m,n}$=1 or not; $x_{m,n}$=0. Lines 4 initializes the tabular matrix (V) while line 5 initializes the $1 \times |\mathcal{M}|$-dimensional vector of zeros and ones, where n value is fixed per MH. Lines 6-14 calculate the optimal CHP value in a content-by-content fashion and get the optimal value Ψ_n^* in line 15. With minor violations to notations, line 16 reassigns the temporary variables. Lines 18-27 conclude the algorithm by backtracking, from matrix V, the content assignment $x_{m,n}$ for tagged n.

Algorithm 2: Optimal solution for ZoSKP

Input: $\mathcal{M}, \mathcal{P}, \mathcal{S}, L_n$
Output: $\Psi_n^*, \mathbf{x}$
Function: DP-ZoSKP($\mathcal{M}, \mathcal{P}, \mathcal{S}, L_n$)
  $\mathbf{V} = zeros(|\mathcal{M}|, L_n)$;
  $\mathbf{x} = zeros(1, |\mathcal{M}|)$;
  for $1 \leq m \leq |\mathcal{M}|$ **do**
    for $0 \leq j \leq L$ **do**
      if $j \geq s_m$ **then**
        $V[m, j] = max(V[m-1, j], \rho_m + V[m-1, j-s_m])$;
      else
        $V[m, j] = V[m-1, j]$;
      end
    end
  end
  $\Psi_n^* = V[|\mathcal{M}|, L_n]$;
  $m = |\mathcal{M}| + 1, j = L_n + 1$;
  $temp = V[m, j]$;
  while $m > 0$ *&&* $j > 0$ **do**
    if $temp \neq V[m-1, j]$ **then**
      $x_{m,n} = 1$;
      $j = j - s_m$;
      $m = m - 1$;
      $temp = V[m, j]$;
    else
      $m = m - 1$;
    end
  end
  return Ψ_n^*, $\mathbf{x} \leftarrow x_{m,n}$
end

4.5.2 Exact solution for ZoMKP formulation

In this subsection, we present the proposed BaB-based strategy, an enumerative and iterative content selection procedure, for the cluster-wide ZoMKP formulation where the caches of all MH are filled progressively. For that purpose, there are a few global input parameters such as the $\{L_n\}$ that refers to the list of free cache sizes in all MHs, and $\{D_n\}$ that denotes the array (*i.e.*, temporary stack) of cache decision vectors taken by up to n^{th} MHs, $n \in \{1, ..., N\}$. The temporary allocation matrix $\hat{x}$, defined as: $\hat{x} = \{\hat{x}_{m,n} : \forall f_m \in \mathcal{M}, n \in \mathcal{N}\}$, is a two-dimensional array of size $|\mathcal{N}| \times |\mathcal{M}|$ that indicates whether content f_m is allocated to the cache of n during the recursion time.

The strategy is detailed in a sequence of subtopics such that the upper performance bound is calculated using Algorithm 3 whereas the lower performance bound is estimated using Algorithm 4. The calculation of both upper and lower performance bounds are found by applying ZoSKP (*i.e.*, Algorithm 2). The main body of the placement strategy is displayed in Algorithm 5, which combines all these algorithms.

Upper bound calculation

The upper bound of caching at multiple caching edges is assumed to be the maximum CHP gain by the cache sizes of the sum of all MHs. Algorithm 3 is used to calculate the UB performance, conditioned on the placement of a given subset of video contents in $\mathcal{M}$, and aggregate of available free spaces. This algorithm is frequently called in intermediate steps of the BB-ZoMKP strategy to evaluate the CHP outcomes of each branch, with a subset of contents marked in $\hat{x}$. Also, the identifier of the current MH (i) is passed as an input to calculate its free cache sizes; by this step, the proposed strategy fills only part of the target i.

Algorithm 3: CHP Upper bound calculation

1 **Input:** $\mathcal{M}, \mathcal{P}, \mathcal{S}, \mathcal{N}, \{L_n\}, \{\mathcal{D}_n\}, i, \hat{x}$
2 **Output:** Ψ_U
3 **Function:** `UB-MKP`$(\mathcal{M}, \mathcal{P}, \mathcal{S}, \{L_n\}, \{\mathcal{D}_n\}, i, \hat{x})$
4 $\quad \bar{L} = \sum_{n=i}^{|\mathcal{N}|} L_n - \sum_{f_m \in \mathcal{D}_i} s_m \cdot \hat{x}_{i,m}$;
5 $\quad \bar{\mathcal{M}} = \{f_m : \forall f_m \in \mathcal{M}, \hat{x}_{m,n} = 0 \text{ for } n = 1, ..., i\}$;
6 $\quad \bar{\mathcal{P}} = \{\rho_m : \forall \rho_m \in \mathcal{P}, \hat{x}_{m,n} = 0 \text{ for } n = 1, ..., i\}$;
7 $\quad \bar{\mathcal{S}} = \{s_m : \forall s_m \in \mathcal{S}, \hat{x}_{m,n} = 0 \text{ for } n = 1, ..., i\}$;
8 $\quad [\bar{\Psi}_U, \sim] =$ **DP-ZoSKP** $\left(\bar{\mathcal{M}}, \bar{\mathcal{P}}, \bar{\mathcal{S}}, \bar{L}\right)$;
9 $\quad \Psi_U = \bar{\Psi}_U + \sum_{n=1}^{i} \sum_{f_m \in \mathcal{D}_n} \hat{x}_{m,n} \cdot \rho_m$;
10 $\quad$ **return** Ψ_U
11 **end**

In Algorithm 3, at Line 4, it calculates the available cache size of MHs that have not yet been examined (*i.e.*, helpers $i+1, ..., |\mathcal{N}|$), plus the residual cache size of helper i following the placement decisions of previous iterations, depicted by input $\hat{x}_{m,n}$ values. In line 5, the algorithm identifies the set of free videos that have not been fixed to MHs (including i) in previous iterations. At line 6 and 7, the respective popularity and sizes of videos in $\bar{\mathcal{M}}$ are filtered that have not yet been placed in any MH.

After calculating the free cache space and non-cached contents $\bar{\mathcal{M}}$, together with the respective sets of content popularity $\bar{\mathcal{P}}$ and sizes $\bar{\mathcal{S}}$, Algorithm 3 deploys the ZoSKP method (Algorithm 2). Mind that each time, allocation of contents from $\bar{\mathcal{M}}$ into a cache of $\bar{L}$ is considered as a single knapsack problem, as shown in line 8. At line 9, the algorithm calculates the current achievable ultimate upper bound of the ZoMKP formulation by including the sum of CHPs of all placement decisions indicated by $\hat{x}_{m,n}$, $\hat{x}_{m,n} \in \hat{x}$. To this end, line 9 adds: i) the maximum CHP that can be obtained from the

ZoSKP subproblem with $\bar{\mathcal{M}}$ and $\bar{L}$ inputs to, ii) the CHP value from the placement decision performed in previous iterations.

Lower bound calculation

The calculation of the lower bound (Ψ_L) is based on sequentially solving about $|\mathcal{N}|$ independent ZoSKP subproblems for each MH, starting from the lowest size. Technically, it calculates the LB value by summing up all optimal CHP gains from individual allocations until i and predicted gains at the unfilled MHs. The performance gap between the UB and LB is because, for the case of UB, the individual constraint in (4.2b) is relaxed to a single cache size constraint, which is simply the surrogate sum of all caches of the MHs (supposed to be similar with 3.2b).

Algorithm 4: Lower CHP bound calculation

1 **Input:** $\mathcal{M}, \mathcal{P}, \mathcal{S}, \mathcal{N}, \{L_n\}, \{\mathcal{D}_n\}, i, \hat{x}$
2 **Output:** $\Psi_L, \tilde{x}$
3 **Function:** LB-MKP$(\mathcal{M}, \mathcal{P}, \mathcal{S}, \{L_n\}, \{\mathcal{D}_n\}, i, \hat{x})$
4 $\quad \bar{\Psi}_L = \sum_{n=1}^{i} \sum_{f_m \in \mathcal{D}_n} \rho_m \cdot \hat{x}_{m,n}$;
5 $\quad \hat{\mathcal{M}} = \{f_m : f_m \in \mathcal{M}, \hat{x}_{m,n} = 0, \text{for } n = 1, ..., i\}$;
6 $\quad \bar{\mathcal{M}} = \hat{\mathcal{M}} \setminus \mathcal{D}_i$;
7 $\quad \bar{\mathcal{P}} = \{\rho_m : \rho_m \in \mathcal{P} \text{ and } f_m \in \bar{\mathcal{M}}\}$;
8 $\quad \bar{\mathcal{S}} = \{s_m : s_m \in \mathcal{S} \text{ and } f_m \in \bar{\mathcal{M}}\}$;
9 $\quad \bar{L} = L_i - \sum_{f_m \in \mathcal{D}_i} s_m \cdot \hat{x}_{i,m}$;
10 $\quad$ **for** n=i *to* $|\mathcal{N}|$ **do**
11 $\quad\quad \left[\tilde{\Psi}_L, \mathbf{y}\right] =$ **DP-ZoSKP**$(\bar{\mathcal{M}}, \bar{\mathcal{P}}, \bar{\mathcal{S}}, \bar{L})$;
12 $\quad\quad \Psi_L = \bar{\Psi}_L + \tilde{\Psi}_L$;
13 $\quad\quad$ **for** $m = 1$ *to* $|\mathcal{M}|$ **do**
14 $\quad\quad\quad \tilde{x}_{m,n} = y_m$;
15 $\quad\quad$ **end**
16 $\quad\quad \bar{\mathcal{M}} = \hat{\mathcal{M}} \setminus \{f_m : \tilde{x}_{m,n} = 1\}$;
17 $\quad\quad \bar{\mathcal{P}} = \{\rho_m : \rho_m \in \mathcal{P} \text{ and } f_m \in \bar{\mathcal{M}}\}$;
18 $\quad\quad \bar{\mathcal{S}} = \{s_m : s_m \in \mathcal{S} \text{ and } f_m \in \bar{\mathcal{M}}\}$;
19 $\quad\quad \bar{L} = L_{n+1}$;
20 $\quad$ **end**
21 $\quad$ **return** $\Psi_L, \tilde{x} \leftarrow \tilde{x}_{m,n}$
22 **end**

Algorithm 4 implements the procedure above to calculate a tight lower CHP performance bound, including the gain by previously allocated contents- indicated in matrix $\hat{x}$. After declaring the input and output variables, it starts at line 4 to evaluate the existing cluster-wide CHP, which is the result of allocating contents into all caches up to the current MH i- which is partially filled. In line 5, the algorithm identifies the set of contents that have not yet been assigned to any MH (*i.e.*, $1, ..., i$). At line 6, free contents from the previous iteration that did not improve the LB when examined for the tagged

MH (*i.e.*, not stacked in D_n) are uselessly excluded from the next iteration. For the next iteration, also the content popularity (line 7) and size (line 8) vectors and the available cache at the current MH (line 9) are filtered.

From lines 10-20, the algorithm progressively adds the CHP gain ($\tilde{\Psi}_L$) from assigning contents to free MHs, starting from available space at the current MH. In more detail, lines 11-12 solve the ZoSKP problem for the residual cache size of i and add the CHP with the cluster-wide lower bound ($\bar{\Psi}_L$). At line 14, the placement decision vector (y_m) for the residual cache are copied to $\tilde{x}_{m,n}$, where $y_m \in \mathbf{y}$. Starting from line 15-20 the input parameters $\bar{\mathcal{M}}$, $\bar{\mathcal{P}}$, and $\bar{\mathcal{S}}$ are updated while $\bar{L}$ is set to next MH for its iteration. The content selection decision after running Algorithm 4 are returned in a two-dimensional matrix $\tilde{x}=\{\tilde{x}_{m,n} : \forall f_m \in \mathcal{M}, n \in \mathcal{N}\}$, which is used in Algorithm 5.

BaB-ZoMKP content placement method

The main recursive enumeration of the proposed content placement strategy to solve the ZoMKP formulation is detailed in this subsection. The thematic aspect of the strategy is that, per each iteration, it evaluates whether a free content f_m can be cached to the current MH. Contents that have been decided for n^{th} MH are stacked into respective data structure D_n. The two-dimensional indicator matrix ($\hat{x}$) is used to store the recurring (might be partial) cache solutions of the ZoMKP instances, where $\hat{x}_{m,n}$=1 if content m is temporarily fixed to MH n and $\hat{x}_{m,n}$=0, otherwise.

The main function of the BB-ZoMKP strategy is an advanced bound-and-bound search mechanism, so we termed this caching policy as the 'bound-and-bound 0/1-Multiple Knapsack Problem' (BB-ZoMKP) strategy, whose pseudocode is presented in Algorithm 5. The BB-ZoMKP consists of four blocks: *Initialize* (lines 5-8), *BaselinePlacement* (lines 10-24), *ContentPlacement* (lines 26-40) and *Backtrack* (lines 42-56). While the BB-ZoMKP selection strategy evaluates for n^{th} MH, caches of $1, ..., n$-1 are assumed to be filled, but caches of $n, ..., |\mathcal{N}|$ are empty. The recursion frequently calculates the LB and UB by calling respective functions to omit content search branches that lead to lower performance bounds.

The *Initialize* block sets the global parameters (lines 5-6) and calculates the ideal upper CHP bound (Ψ_{UB}), at line 7, over the original content library $\mathcal{M}$ and total size of: $L = \sum_{n=1}^{|\mathcal{N}|} L_n$, using the optimal ZoSKP allocation in Algorithm 3. This UB value is the ideal maximum CHP performance, so we call it *tightest* (Ψ_{UB}) because it is estimated when all MHs are initially free. In contrast, other consecutive upper bounds (Ψ_U) are calculated while some MHs are filled with contents, where unusable residual spaces will be left behind. Therefore, the tightest Ψ_{UB} performance bound is stored (line 8) for future comparison. It is worth noting that other blocks never call back to the *Initialize*, even though they make extensive inter-block looping.

The *BaselinePlacement* block is responsible for calculating the lower bound (Ψ_L) and evaluates the performance of each branch recursion. It decides whether LB is improved, approaching the Ψ_{UB}, and defines the corresponding placement decision $\tilde{x}$ (line 10), given a problem instance where the list of free MHs' to be filled. It takes input parameters of current helper i, the stack (D_n) of previously fixed contents, and the

Algorithm 5: BaB Caching Strategy

Input: $\mathcal{M}, \mathcal{P}, \mathcal{S}, \mathcal{N}, \{L_n\}$
Output: $\Psi^*_{\mathcal{C}}$, $\mathbf{x}$
Function: `BB-ZoMKP`$(\mathcal{M}, \mathcal{P}, \mathcal{S}, \{L_n\})$
  [Initialize]
  for $n = 1$ *to* $|\mathcal{N}|$ **do** $\mathcal{D}_n = \emptyset$ **end**;
  $\hat{x} = zeros(N, |\mathcal{M}|)$; $\psi = 0$; $i = 1$;
  $\Psi_U =$ **UB-MKP**$(\mathcal{M}, \mathcal{P}, \mathcal{S}, \{L_n\}, \{\mathcal{D}_n\}, i, \hat{x})$;
  $\Psi_{UB} = \Psi_U$;
  [BaselinePlacement]
  $[\Psi_L, \tilde{x}] =$ **LB-MKP**$(\mathcal{M}, \mathcal{P}, \mathcal{S}, \{L_n\}, \{\mathcal{D}_n\}, i, \hat{x})$;
  if $\Psi_L > \psi$ **then**
    $\psi = \Psi_L$; $\mathbf{x} = \hat{x}$;
    for $n = i$ *to* $|\mathcal{N}|$ **do**
      for m=*1 to* $|\mathcal{M}|$ **do**
        if $\tilde{x}_{m,n} = 1$ **then** $x_{m,n} = 1$; **end**
      end
    end
    if $\psi = \Psi_{UB}$ **then**
      $\Psi^*_{\mathcal{C}} = \Psi_{UB}$; **return** $\Psi^*_{\mathcal{C}}, \mathbf{x} \leftarrow x_{m,n}$
    end
    if $\psi = \Psi_U$ **then**
      Go to **Backtrack**;
    end
  end
  [ContentPlacement]
  repeat
    $\mathcal{F} = \{\delta : \tilde{x}_{i,\delta} = 1\}$;
    while $\mathcal{F} \neq \emptyset$ **do**
      $m = \min\{\delta : \delta \in \mathcal{F}\}$;
      $\mathcal{F} = \mathcal{F} \setminus \{m\}$;
      $\mathcal{D}_i = Push(\mathcal{D}_i, f_m)$;
      $\hat{x}_{i,m} = 1$;
      $\Psi_U =$ **UB-MKP**$(\mathcal{M}, \mathcal{P}, \mathcal{S}, \{L_n\}, \{\mathcal{D}_n\}, i, \hat{x})$;
      if $\Psi_U \leq \psi$ **then**
        Go to **Backtrack**;
      end
    end
    $i = i + 1$;
  until i=N;
  $i = i - 1$;
  [Backtrack]
  repeat
    while $\mathcal{D}_i \neq \emptyset$ **do**
      Let m be content on top of $\mathcal{D}_i$;
      if $\hat{x}_{i,m} = 0$ **then**
        $\mathcal{D}_i = Pop(\mathcal{D}_i, f_m)$
      else
        $\hat{x}_{i,m} = 0$;
        $\Psi_U =$ **UB-MKP**$(\mathcal{M}, \mathcal{P}, \mathcal{S}, \{L_n\}, \{\mathcal{D}_n\}, i, \hat{x})$;
        if $\Psi_U > \psi$ **then**
          Go to **BaselinePlacement**;
        end
      end
    end
    $i = i - 1$;
  until i=*0*;
end

existing allocation vector $\hat{x}$. For a given iteration, if the placement solution found by the LB-MKP algorithm results in improved CHP, compared to the previous LB (ψ), then the ψ value is updated to Ψ_L (line 12). In addition, the final decision indicator matrix (x) is updated from the previous allocation indicator ($\hat{x}$) and from the new placement decision matrix ($\tilde{x}$) of the LB-MKP function (lines 13-17), the last case is for $n \geq i$.

Accordingly, if the updated LB performance matches with the tightest Ψ_{UB}, the BB-ZoMKP strategy concludes the selection process (lines 18-20). Thus, it returns the exact cluster-wide CHP gain (Ψ_C^*) and its final caching decision table (x), that contains all $x_{m,n}$ values. But if the updated LB does not match the tightest Ψ_{UB} and still has free MHs to asses, the selection strategy continues to examine other instances. Note that after *Initialize* block, it holds $\Psi_U \leq \Psi_{UB}$ so that the next 'if-condition' (lines 21-23) refers to only recursion that came from the *Backtrack* block. Hence, it checks whether the updated LB and current UB match. If so, the *Backtrack* of the strategy is called. This step enables the strategy to search for other instances that improve the upper bound Ψ_U and explore solutions with higher CHP gains (*i.e.*, closer to the tightest Ψ_{UB}). If the two conditions do not happen, the procedure stacks contents to D_n in the next block.

The *ContentPlacement* block of the strategy is the part where the content assignment solution, derived from the LB-MKP function of *BaselinePlacement*, is stacked to D_n and further assessed. In this case, the assignment solution gives the condition of: $\psi \leq \Psi_U \leq \Psi_{UB}$. The respective cache temporary decisions ($\hat{x}$) is updated for the current MH i and repeats for all MHs, in the ascending order of cache size (lines 26-39). Note that the contents are already in the order of their 'value per unit size' so that contents are pushed to the stack, starting from the highest efficiency. Each time a content f_m is stacked, its impact on the UB is checked (line 33). If it is lower than the current CHP performance ψ (line 34), some of the instances are pruned off using the *Backtrack* block (lines 34-36). The procedure resumes after corrective measures are taken using the Backtrack logic, potentially running recursively part of the strategy again to identify a better content placement solution. It is worth noting that, given $\hat{x}_{m,n}, n = 1, ..., |\mathcal{N}|$-1 is settled, the allocation by the LB-MKP function is optimal. Therefore, we do not need to make backtracking for the last MH (*i.e.*, it is done starting from $i = |\mathcal{N}|$-1, line 40).

The *Backtrack* block of the strategy is called under occasions when the selected solution either does not improve the performance bounds of the previous allocation ($\psi = \Psi_U$) or it not giving a feasible solution (*i.e.*, $\Psi_U \leq \psi$) after an iteration. In such cases, its main role becomes pruning off such allocations that deteriorate the performance UB of the conditional (partial) solution to the problem as depicted by the allocations D_n. To this end, it starts removing contents from the MHs' cache (line 48) that triggered the Backtrack call and then evaluates if the upper bound CHP performance is improved through this action (line 49). If not, the Backtrack recursion continues to remove contents from the D_n of the current MH (line 55) and also may continue to other MHs (line 56) until the old feasible instances are traced back. In this case, a new *BaselinePlacement* is triggered to make other feasible allocations (lines 50-52). The BB-ZoMKP content placement strategy terminates either when a solution is shown to attain the tightest CHP probability Ψ_{UB} (lines 20), or when no further improvements can be achieved on the CHP ψ attained by the current solution x (lines 57).

4.6 System-level Simulation and Analysis

In this section, we assess the performance of the proposed BB-ZoMKP content placement strategy and compare it to that of two widely used strategies: the *Random* and the *Greedy* placement strategies, using cluster CHP (Ψ_C) as the KPI parameter. Different variants of the two strategies are found in the state-of-the-art, *e.g.*, for Random in [33], [134]–[136] and for Greedy in [14], [130], [137], [138]. The Random strategy arbitrarily selects content from the library $\mathcal{M}$ of popular contents and places it without repetition in the caches of all MHs belonging to the cluster. The Greedy strategy orders videos in $\mathcal{M}$ in descending order of their popularity and places them in the cache of MHs sequentially, *i.e.*, the cache of the first MH is filled by skipping contents that cannot fit; then, the second MH is filled with the free contents, etc., until no other content can fit into the cache of any MH. For all strategies, the MHs are sorted in ascending order of their cache size and filled by keeping the no-repetition and no-partition policies.

We investigated the performance of the BB-ZoMKP, Random, and Greedy content placement strategies using extensive system-level simulations. To this end, we consider a multi-tier HCN that includes two types of cache-enabled MHs of SBSs and FBSs. Accordingly, we consider a MEC cluster of $\mathcal{N}_1$ sets of SBSs and $\mathcal{N}_2$ FBSs. The cache size of each MH type is modeled by a normal distribution of mean cache size ξ_1 GB with standard deviation σ_1^2, and ξ_2 GB with σ_2^2, respectively. Thus, for a tagged SBS i the cache size is given as $L_i \in \mathcal{L}_1 \simeq norm(\xi_1, \sigma_1^2)$, whereas for a tagged FBS j the cache size $L_j \in \mathcal{L}_2 \simeq norm(\xi_2, \sigma_2^2)$ such that $\mathcal{L}^{1x120} = \{L_n : n = 1, ..., |\mathcal{N}|\} = \mathcal{L}_1 \cup \mathcal{L}_2$. Accordingly, the total cluster cache size is modeled as: $L \simeq |\mathcal{N}_1| \cdot norm\left(\xi_1, \sigma_1^2\right) + |\mathcal{N}_2| \cdot norm(\xi_2, \sigma_2^2)$.

Regarding the library $\mathcal{M}$, we consider that there are $|\mathcal{M}|$ number of video contents, where the size of each video is assumed to follow an exponential distribution of mean size μ GB (*i.e.*, $s_m \in \mathcal{S}^{1x5000} \simeq exp(\mu), \forall f_m \in \mathcal{M}$). This gives us on the average a total size (of all videos) of $E\left[\sum_{m=1}^{|\mathcal{M}|} s_m\right] = |\mathcal{M}| \cdot \mu$. The popularity distribution of video contents in $\mathcal{M}$ is assumed to follow the widely-accepted Zipf distribution according to which, the popularity value ρ_m of the m^{th} most popular content in $\mathcal{M}$ is given by Eq. 2.1, where $\gamma \geq 0$ is the Zipf parameter (popularity skewness) and $f_m \in \mathcal{M}$.

Unless differently stated, the values for the main simulation parameters discussed above are fixed as in Table 5.1. According to the average bit rate values recommended by Google for video uploads in YouTube [123] and the NTT-DOCOMO guideline for video delivery in mobile data networks in [124], the μ depends on the video codec type, the video bit rate and the type of the application (*e.g.*, gaming, music video clips).

TABLE 4.1: CHP numerical parameter values

Parameters	Values	Distribution
Number of SBS (N_1)	20	Fixed
Number of FBS (N_2)	100	Fixed
Mean SBS cache size (ξ_1)	$20 \cdot \xi_2$ GB	Normal
SBS cache size variance (σ_1)	10 GB	Normal
Mean FBS cache size (ξ_2)	10 GB	Normal
FBS cache size variance (σ_2)	2 GB	Normal
Content library size $\lvert\mathcal{M}\rvert$	5000 videos	Fixed
Mean content size (μ)	4 GB	Exponential
Popularity parameter (γ)	1.0	Zipf

TABLE 4.3: Major social media video limits, May 2021.

Social Media Platform	Maximum Video Size (GB)	Maximum Duration	Maximum Resolution
Facebook	10 GB	240 min.	4096x2048
Instagram	4 GB	1 min.	1080x1080
Pinterest	2 GB	30 min.	640x640
Snapchat	1 GB	no limit	1080x1290
YouTube	128 GB	12 hrs.	3840x2160
LinkedIn	5 GB	10 min.	1920x1080

In Table 4.2, we summarize some common YouTube video types and their respective size. In Table 4.3, we overview the limitations adopted by major social media providers on uploading videos to their platforms. According to the values presented in this table, we fix the mean content size value μ to 4 GB, which corresponds to an HDR (4K) YouTube video of display resolution 3840x2160 (2160p) at a High Frame Rate and length of approximately 8 minutes, at bit rate 66 Mbps. Such types of videos currently dominate Internet traffic. When we fix γ=1.0, it corresponds to the case that 10% of videos account for 75% of the popularity.

4.6.1 Impact of the MH mean cache size

We begin the performance evaluation of the strategies under increasing mean SBS cache size ξ_1. Note that in our simulations, we have set the FBS mean cache size ξ_2 to be proportional to the respective ξ_1 as: $\xi_1 = 20 \cdot \xi_2$ (Table 5.1); thus, an increase to the mean SBS cache size proportionally increases both the FBS cache size and the total MEC cluster cache size (*i.e.*, $L \simeq |\mathcal{N}_1| \cdot norm\left(20 \cdot \xi_2, \sigma_1^2\right) + |\mathcal{N}_2| \cdot norm\left(\xi_2, \sigma_2^2\right)$.

TABLE 4.2: Sample content statistics calculated from YouTube video upload recommendation.

Codec type	**Frame Rate**	**Video Bit rate** (Mbps)	**Type of application**	
			Gaming, films, (18.9 min.) average	**VIP people, music videos (aver. 8.2 min.)**
SDR(4K) (2160p)	Standard Frame Rate	35-45	5.67 GB	2.46 GB
	High Frame Rate	53-68	8.6 GB	3.7 GB
HDR(4K) (2160p)	Standard Frame Rate	44-56	7.1 GB	3 GB
	High Frame Rate	66-85	10.7 GB	4.6 GB
Audio bitrate =192 Kbps is considered for all video bit rates				

In Fig. 4.3, we plot the cluster-wide mean CHP for increasing size ξ_1 under three different mean content sizes μ. Recall that the cache size of the SBS i is distributed according to a normal distribution $L_i \simeq norm(\xi_1, \sigma_1^2)$ and that the content size of popular videos follows an exponential distribution $s_m \simeq \exp(\mu)$. We notice that for a fixed library size ($\mathcal{M}$=5000-Table 5.1), increasing the SBS cache size improves the mean CHP value of the cluster for all content placement strategies. The improvement for the Random strategy is linear due to the random utilization of the available cache size, whereas a logarithmic increase is observed for the proposed and Greedy strategies in the case of low ξ_1 values. This indicates that both strategies can better exploit the additional buffer available to the cluster by the SBS MHs.

From the plot, we observe that for a given set of fixed system parameters and a mean content size μ, the performance of the proposed and Greedy strategies is similar when the mean SBS cache size ξ_1 is small (*e.g.*, ξ_1<20 GBs). Nonetheless, the respective performance gap grows rapidly as the SBS cache size increases for all μ values under the scope. The performance improvement attained by the proposed strategy is due to considering the full state space of the ZoMKP content placement tree, where each branch instance fixes a candidate set of contents to a specific MH. This strategy explores the solution tree by moving in depth to assess the CHP performance of every solution branch by the UB and LB limits to eliminate sub-branches with sub-optimal instances, using the backtracking of the current optimal solution. In this fashion, the proposed strategy can smartly explore the full state space and identify the exact, not the only optimal, placement solution. The Greedy performs behind the proposed strategy because it always picks contents with the currently highest popularity and skips to the next one when the file does not fit.

Regarding the combined impact of μ and ξ_1 parameters, Fig. 4.3 illustrates that a

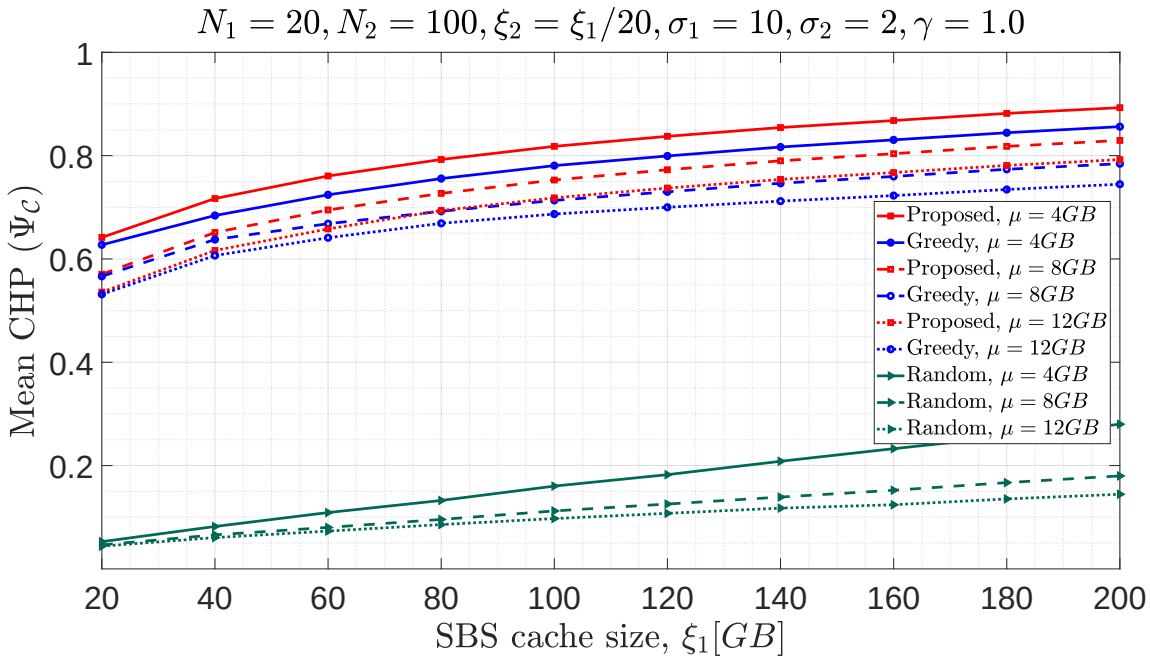

FIGURE 4.3: The CHP outcome for increasing SBS cache size under three content mean sizes.

lower mean content size μ (*e.g.*, μ=4 GB) enables all strategies to attain higher CHP, especially when ξ_1 increases. This directly follows from the fact that for $\mathcal{M}$, a larger cache size increases the probability of all strategies making a massive allocation in the cluster, provided that a larger volume of popular contents can fit in the existing MH cache. Besides, this is why for a lower μ=4 GB, the Random exhibits a rapid increase in its performance as compared to higher (*e.g.*, μ=8, 12). Interestingly, the proposed strategy for μ=12 GB is shown to attain a similar CHP with the Greedy for μ=8 GB, highlighting that the proposed strategy enables an enhanced cache capabilities utilization in the MEC cluster even when larger contents are considered.

The superior performance of the proposed strategy is also highlighted in Fig. 4.4 as well, in terms of available resource utilization, on a cluster-wide scale for different μ values. In Fig. 4.4, the proposed strategy attains a utilization close to 100% in all scenarios under the scope. In contrast, the Greedy and Random strategies are shown to leave a small number of cache resource utilization. Unlike the big CHP performance gap, shown in 4.3, all strategies almost closely utilize the cache resource. This happens regardless of mean content size μ is negligible to the utilization of the available cluster-wide cache size, i.e., the different μ values considered in the legend result in a similar performance for each strategy.

Let us now investigate the impact of a different mixture of cache sizes across the SBS and the FBS on the cluster CHP performance, assuming that the ξ_1 of an SBS MH is proportional to the ξ_2 of FBS, with ratio β; *i.e.*, ξ_1=$\beta \cdot \xi_2$. Fig. 4.5 plots the impact of the SBS/FBS cache size ratio, in the cluster-wide CHP, under different μ values for the FBS tier ξ_2=4, 8, 20 GB. For β=0, the CHP performance is the result of utilizing only the $\mathcal{N}_2$ FBS (i.e., no SBSs since ξ_1=0 GB). As expected, the performance of all

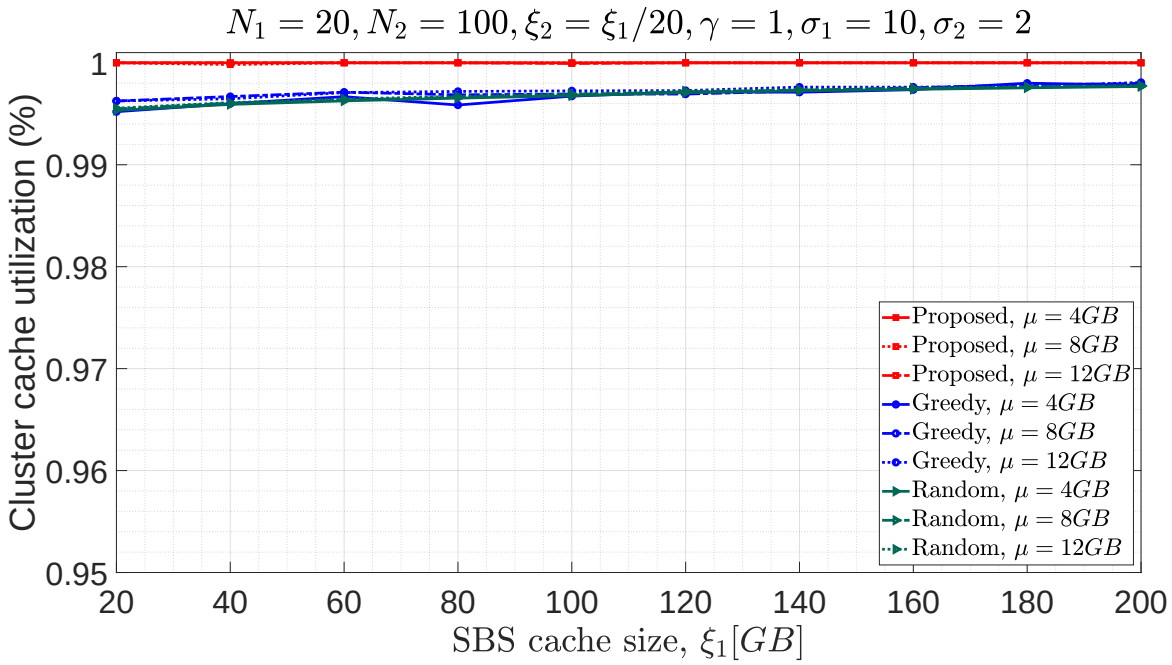

FIGURE 4.4: Percentage of total used cache size for increasing SBS cache size and different content sizes.

content placement strategies increases with ξ_2 and β. In fact, for ξ_2=50 GB and β=50, the available cache size in the MEC cluster is sufficient to store the entire set of video contents in $\mathcal{M}$, enabling all strategies to attain a CHP close to 1.

Similar to Fig. 4.3, the CHP of the Random strategy increases linearly with the available cache size, modeled by the ratio β in Fig. 4.5. However, the ratio of the CHP improvement strongly depends on the mean cache size ξ_2 assumed for the FBS. For β=0 (*i.e.*, no SBS), we observe that the Random performs poorly independent of the μ, whereas the proposed and Greedy strategies exhibit similar performance as far as they are able to exploit the available cache size, fitting popular contents with better popularity (*e.g.*, for ξ_2=4 GB, they may fit at least one popular content on the μ=4 cache). As the β increases and the ξ_2 gets higher, *e.g.*, β>40 and ξ_2=20 GB, the CHP performance of the Random is shown to close fast the CHP gap with the proposed and Greedy strategies. This indicates that random content placement can provide comparable benefits with a more sophisticated and large volume of storage resources, which can be useful in the case of MEC clusters with low processing capabilities.

Nonetheless, the random content placement performs poorly when the MEC cluster consists of SBS with low storage capacity (*e.g.*, β<20). On the contrary, both the proposed and the Greedy strategies exhibit high CHP performance gains even when the available cache size at the FBS is low (*e.g.*, ξ_2=4 GB) and when the cache size of the SBS is large (*e.g.*, β>20). In both cases, the proposed content placement strategy outperforms the Greedy, enabling the MEC cluster to attain 3-5% better CHP performance, especially when the total cluster size cannot support caching of a large volume of popular video contents (*i.e.*, CHP is lower than 1).

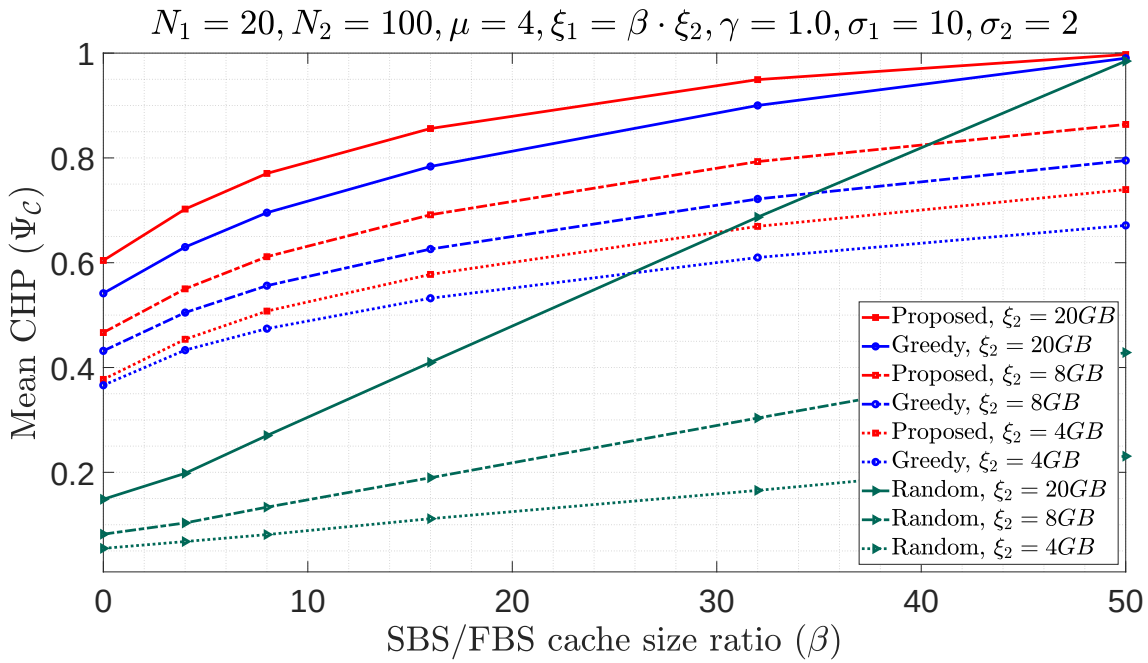

FIGURE 4.5: The CHP outcome for a different mixture of SBS and FBS MHs' cache size in the MEC cluster.

In Fig. 4.6, we investigate the interplay between increasing the mean SBS cache size under different popularity distribution skewness levels. Interestingly, the performance of the Random is shown to remain unaffected by the popularity parameter γ. This follows from the fact that random content placement, the average of thousands of simulations, takes contents from any corner of the Zipf distribution- independent of skewness level. The plot also illustrates that with a lower popularity parameter γ (*e.g.*, by comparing γ=1.2 to γ=0.4), the performance of all strategies deteriorates.

However, the performance gap between the proposed and the Greedy strategies increases fast as the value of γ is decreased (*e.g.*, compare γ=0.8 and γ=0.4). This is because a lower popularity skewness γ spreads the popularity to more content leading to a large number of contents with comparable popularity but different size. Provided that the Greedy strategy is size-agnostic and assigns popular video contents based on only the popularity of the contents in $\mathcal{M}$, it is clear that popularity-based greedy content placement is unable to infer an appropriate combination of video contents to each MHs. This is the result of the heterogeneity in content size and MH cache size.

On the contrary, the proposed strategy better adapts to the heterogeneity of both the content sizes in $\mathcal{M}$ and the cache size of the cluster, enabling enhanced performance. In particular, when the popularity is spread to a larger number of videos (*i.e.*, low skewness index like γ=0.4) and the cluster cache size is more diverse (*e.g.*, for this case, when ξ_1=100 GB, we have 95 GB more cache available in the SBSs compared to ξ_2 of the FBS), the performance gap between the proposed and the Greedy strategies largely increases. For ξ_1=100 GB, we observe that the respective performance gap reaches up to a 20% CHP improvement on an absolute scale and 40% on a relative scale.

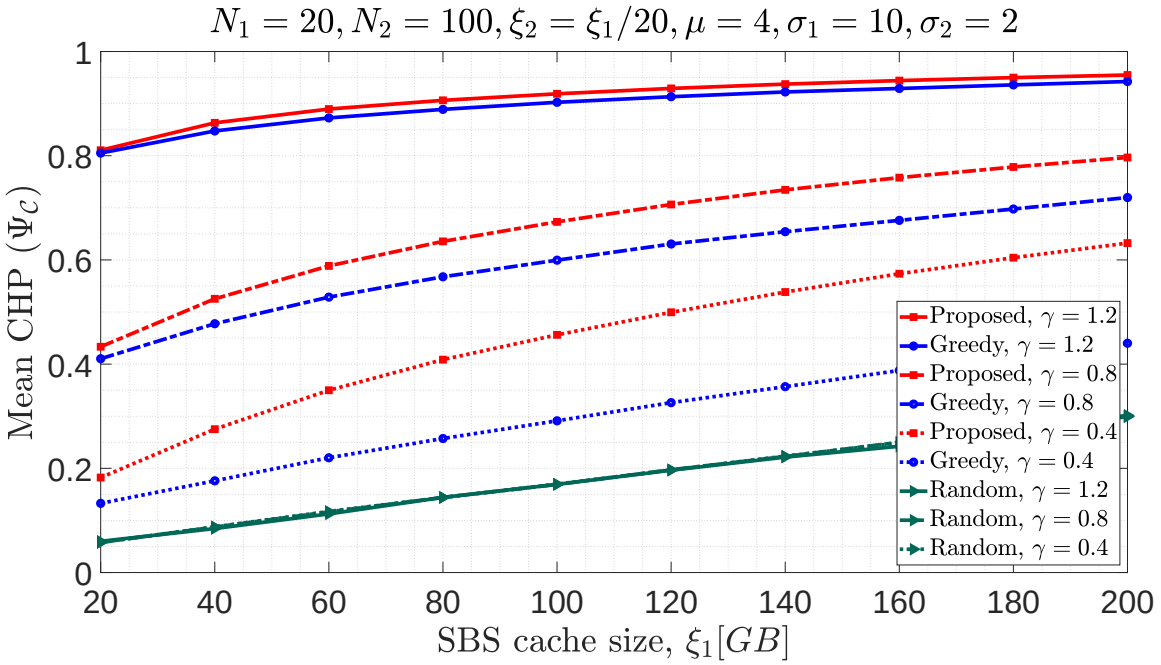

FIGURE 4.6: The CHP outcome for increasing SBS cache size and Zipf popularity skewness.

As a takeaway result, we conclude that the proposed BB-ZoMKP strategy can better exploit the cache resource of the MEC clusters with higher heterogeneity in terms of content size and cache size per MH, enabling the MEC cluster to attain a higher CHP performance. This performance improvement is even more evident when the content popularity is spread over a large list of video contents (*e.g.*, when the Zipf parameter γ is lower). Taking into consideration that the performance of the proposed and Greedy strategies is comparable for a higher value of γ, such as γ>1.2, another interesting observation is that the MEC cluster head may choose to run the Greedy strategy instead of the proposed one to optimize computational resources.

4.6.2 Impact of the content popularity

Given a list of system parameters, in Fig. 4.7, we assess the impact of the popularity distribution γ parameter on the CHP performance under different μ. As an independent variable, the popularity skewness parameter ranges from γ=0 (which means that the contents have uniform popularity) to γ=2 (which means that only a few contents account for the very large portion of the popularity value). For example, in our case of $|\mathcal{M}|$=5000, for γ=0.2 indicates that 1% of contents account for 2.8% popularity) whereas for γ=2 only 1% contents account for about 98.8% of popularity. Similar to Fig. 4.6, we observe that γ has a negligible impact on the performance of the Random strategy but with small deterioration for smaller μ, such as 3 GB, due to the fact that while a very small number of videos in $\mathcal{M}$ concentrate high popularity (*i.e.*, high γ values), the Random strategy selects a subset of contents with lower popularity (*i.e.*,

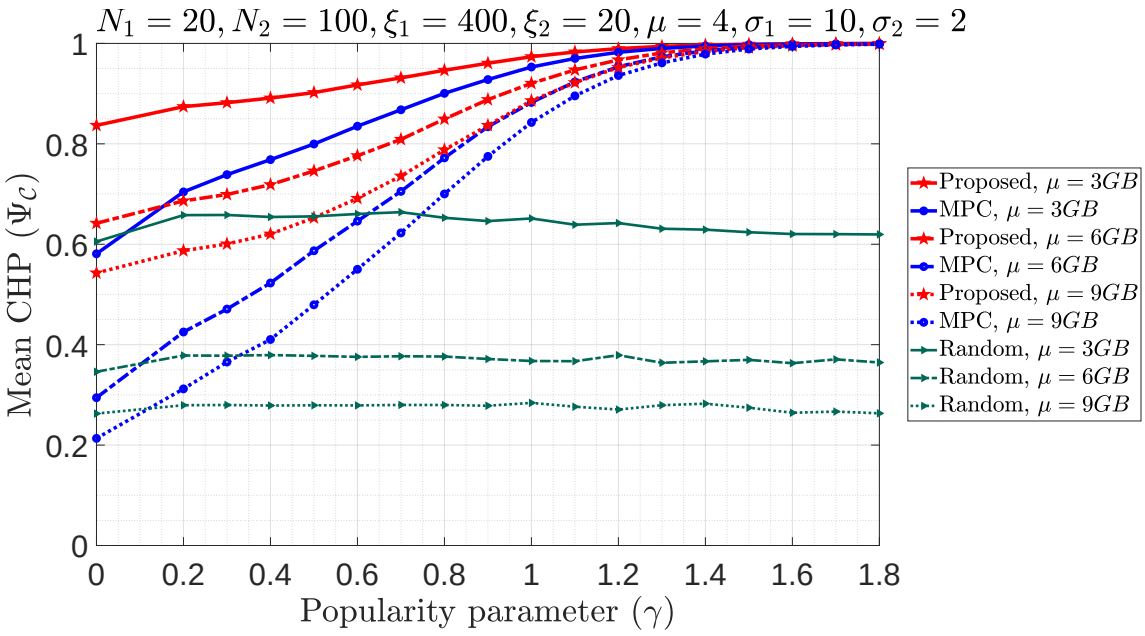

FIGURE 4.7: The CHP outcome for increasing popularity skewness under different mean content sizes.

the most popular, but a few contents, will be dominated by least popular contents so the Random strategy favors for this larger number of contents).

On the contrary, the performance of both the proposed and the Greedy strategies amass cluster CHP with the increase in popularity parameter γ. More interestingly, we get nearly Ψ_C=1 for γ>1.4 under the fixed system parameters. This performance trend is due to the fact that a higher γ parameter translates to the concentration of the popularity in $\mathcal{M}$ to a smaller volume of video contents, enabling popularity-aware strategies to fill the caches of the MEC cluster by including those highly popular contents and leaving unpopular contents does not affect the result. Another interesting observation is that for the proposed and Greedy strategies, the impact of the popularity parameter γ on the CHP performance strongly depends on the mean cache size μ for low popularity values γ (*e.g.*, γ<0.8). This readily follows that even though the popularity is evenly distributed, as the μ decreases, the number of cached contents increases so that the CHP value increases in the cluster. From the same plot, we observe that for low popularity values of γ, the proposed strategy attains the highest CHP gains as compared to the Greedy strategy. On the other hand, for high γ, the CHP performance of the Greedy and the proposed strategies are comparable. This trend is useful in scenarios where the MEC cluster head identifies that the popularity of contents concentrates on a small number of video contents, enabling it to save processing resources by employing greedy methods instead of the proposed (exact) one.

The proposed strategy is shown to outperform the Greedy strategy in terms of CHP probability, especially when the content size of popular videos is large. For example, for μ=9 GB, the proposed strategy is shown to double the Greedy strategy, in terms of the CHP, when the Zipf parameter is very low (*e.g.*, γ=0.2). Once again, the proposed

strategy is shown to better handle the cache available in the MEC cluster even for larger video contents, which is observed that the CHP of the proposed strategy for μ=9 GB is higher than the CHP performance of the Greedy strategy for μ=6 GB, for all γ.

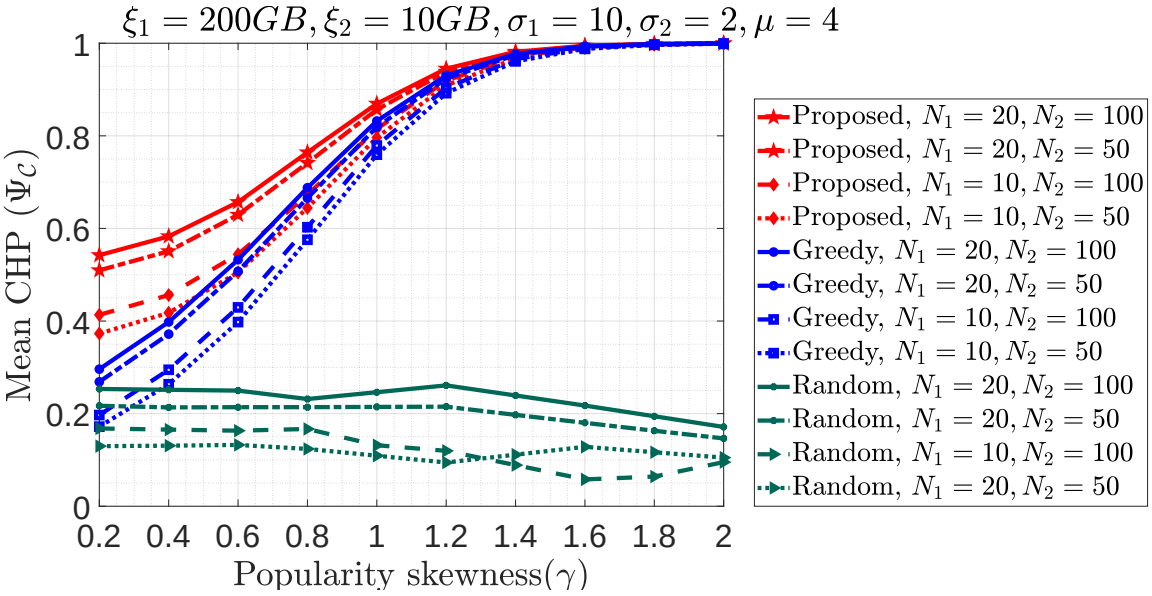

FIGURE 4.8: The CHP outcome for increasing popularity skewness and different number of SBS and FBS.

In Fig. 4.8, we investigate the impact of the popularity skewness γ under a different number of SBS and FBS MHs. Note that as the number of SBS and FBS MHs increases, the available cluster cache size is considered to increase proportionally. Similar to Fig. 4.7, for all strategies, a two-fold increase in the number of large-sized MHs (*i.e.*, with large cache capacity like SBS) is shown to provide significantly better CHP gain as compared to the gain attained by doubling the number of small-sized MHs (*i.e.*, MHs with small cache capacity like FBS). This is because SBSs have 20x higher cache capacity than FBSs, enabling the strategies to handle lots of popular contents.

The CHP performance gap between the proposed and the Greedy strategies is also seen decreasing fast as γ increases for all MH heterogeneity scenarios. In Fig. 4.8, increasing the number of large-sized MHs can play a key role when the popularity of video contents is spread across $\mathcal{M}$. But adding more MHs will have a lower impact on the CHP performance when the popularity is concentrated in a smaller number of contents (*i.e.*, for higher γ values). This performance trend can be used to adjust the MEC cluster size dynamically in view of characterizing the popularity distribution in the library of videos. With this regard, a MEC cluster with a smaller number of large-sized cooperative MHs is needed when lesser spatial diversity in content popularity is observed. On the other hand, if a lower amount of contents account for higher popularity, but the popularity distribution changes fast across the geographical areas, the MNO can cluster a larger number of smaller-sized MHs.

Another interesting observation is that the proposed content placement strategy can better adapt to the existence of a limited number of MHs in the MEC cluster while

attaining a similar performance to the Greedy strategy, with even much better gain for smaller γ values. In particular, we observe that the proposed strategy for $|\mathcal{N}_1|$=10 attains better (for γ<0.6) or comparable performance with that of the Greedy strategy for $|\mathcal{N}_2|$=20 (a double-fold increase of large-sized MHs).

4.6.3 Impact of the mean content size

In this subsection, we analyze the impact of content mean size (μ) under two cases: i) varying γ and, ii) cache size heterogeneity, as shown in Fig. 4.9 and Fig. 4.10, respectively. For this purpose, while other necessary parameters are fixed, the mean size ranges from μ=2 GB to μ=12 GB.

As seen from Fig. 4.9, when μ is larger, the cluster CHP result deteriorates for all strategies. This mainly follows from the fact that the available cache size, per each MH and also cluster-wide, remains fixed while the content size bursts out. We observe a roughly linear performance deterioration for all strategies above the mean content size μ=4 GB value because the number of fitting contents decreases.

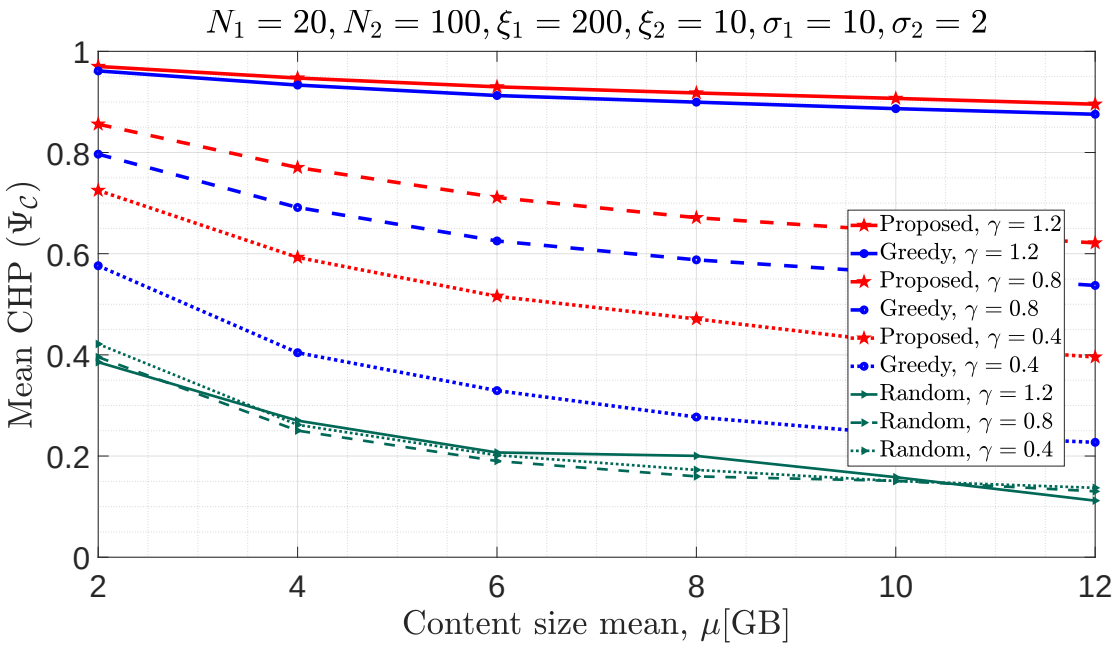

FIGURE 4.9: The cluster CHP for increasing content mean size under different popularity skewness.

The performance deterioration is shown to strongly depend on the popularity distribution. For different γ values plotted in Fig. 4.9, we observe that the performance of the Random strategy remains almost unaffected by γ, while a constant increase of the γ (*e.g.*, from 0.4 to 0.8, or 0.8 to 1.2), increases the CHP performance of both the proposed and Greedy strategies. This happens in a constant rate (*e.g.*, for μ=4 the CHP of the proposed increases by roughly 20% when γ increases by 0.4). The performance gap between the proposed and Greedy strategies increases with μ, highlighting that the proposed strategy can better cope with larger contents as compared to the Greedy.

The CHP performance of caching strategies in the case of increasing μ is impacted by the cache size heterogeneity, as shown in Fig. 4.10. For all spans of content mean size, having large-sized SBS is preferable to having many small-sized FBS. From this plot, we observe that the impact of the number of FBS is not significant on the CHP performance of both the proposed and the Greedy strategies when a large number of large-sized SBSs are considered (*i.e.*, $|\mathcal{N}_1|$=20). However, when the number of large-sized MHs is lower ($|\mathcal{N}_1|$=10), the respective CHP improvement observed by increasing two-fold the number of small-sized MHs is significant for both strategies. On the other hand, for the Random strategy, we observe that the CHP performance shows notable improvement with a two-fold increase of either the SBS or the FBS helpers. This improvement is shown to be similar between the two scenarios (*i.e.*, a two-fold increase of $|\mathcal{N}_1|$ results in similar improvements with a two-fold increase of $|\mathcal{N}_2|$).

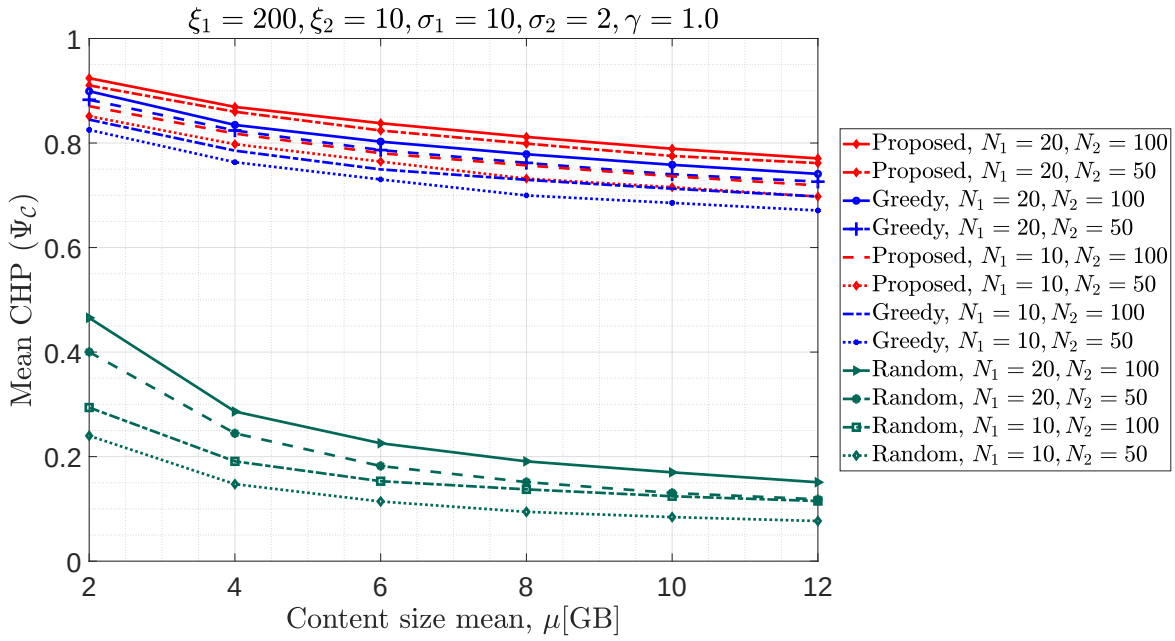

FIGURE 4.10: The CHP outcome for increasing content size and different numbers of SBSs and FBSs.

Interestingly, the impact of increasing $|\mathcal{N}_1|$ and $|\mathcal{N}_2|$ is roughly independent of content mean size for the proposed and Greedy placement strategies, and they show uniform performance gaps. This performance trend indicates that increasing the number of MHs in the MEC cluster will result in the same CHP improvement, independent of the mean content size μ characterizing the contents in $\mathcal{M}$. A similar situation happens to the Random for increasing $|\mathcal{N}_2|$; however, the performance gap between $|\mathcal{N}_1|$=10 and $|\mathcal{N}_1|$=20 declines when μ gets larger due to its poor handling to the available cache.

4.6.4 Impact of the number and type of cache-enabled MHs

In this subsection, we analyze the impact of the number of caching edges, on the content placement performance, by fixing cache mean sizes of ξ_1=200 GB, ξ_2=10 GB, and

cache size normal distribution variances: σ_1=10, σ_1=2, with content mean size μ=4 GB.

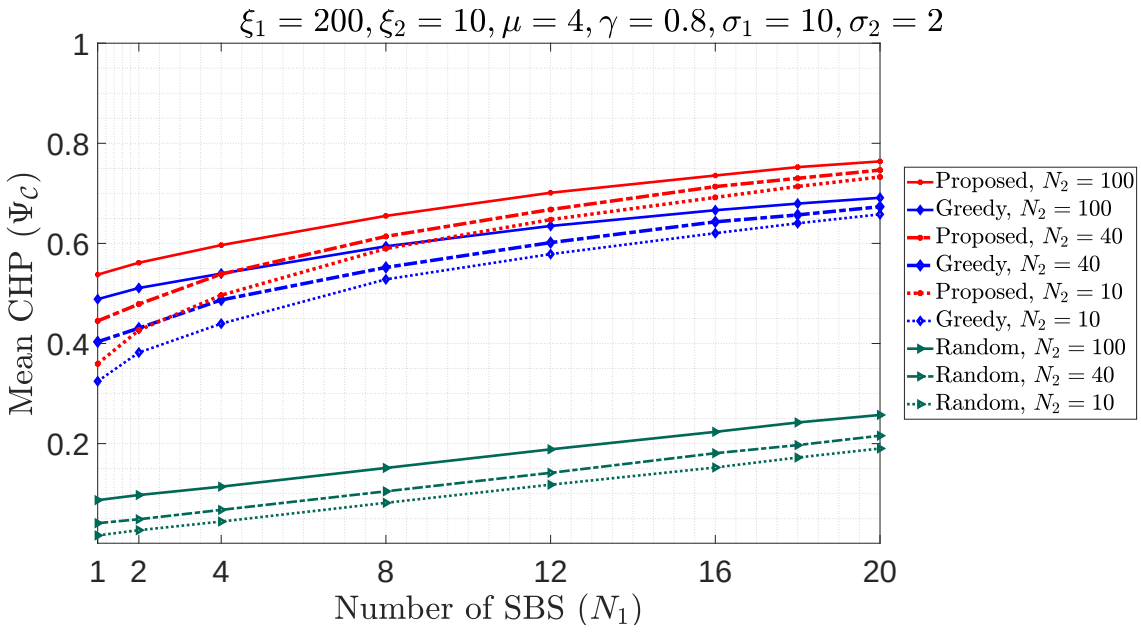

FIGURE 4.11: The CHP outcome for an increasing number of SBSs under a set of FBSs without fixing cluster cache size.

As can be seen in Fig. 4.11, increasing the number of larger-sized MHs, means the SBS improves the performance of the strategies because more caching space is available. Here the main observation is that while there are few SBSs (such as $|\mathcal{N}_1|$=1), the impact of increasing the number of FBSs (such as the increment $|\mathcal{N}_1|$=40 to 100) gives a significant CHP gain. This case gets narrowed when the number of SBSs is high, such as $|\mathcal{N}_1|$=20, for both the proposed and the Greedy strategies. This performance trend highlights that the addition of only a few large-sized MHs has a high added value on the CHP performance, especially when the size of the MEC cluster is small.

A similar performance trend is observed in all caching strategies for increasing $|\mathcal{N}_2|$. However, as the $|\mathcal{N}_1|$ value increases, the proposed and Greedy strategies are shown to improve the CHP at a lower rate for the same increase of $|\mathcal{N}_2|$, which is not the case for Random strategy. For example, while $|\mathcal{N}_1|$=2, we observe a greater CHP improvement for change of $|\mathcal{N}_2|$=10 to 40 as compared to that obtained for $|\mathcal{N}_1|$=16 with the same $|\mathcal{N}_2|$ increase. This performance trend is useful, from the cluster design perspective, to have a higher number of large-sized MHs rather than small-sized MHs. To that extent, the MNO has to trade-off between cache edge densification by having multiple FBS and performance gain, which is examined given the current mixture of MHs to maximize the added value following the formation of a larger cluster.

4.6.5 Impact of the library size of contents

In this subsection, we inspect the impact of having a number of popular contents in the system, which indirectly shows the geographical content usage statistics. In some areas

such as modern cities, a huge number of early-adopting users request an overwhelming range of contents while in other areas, such as underdeveloped regions, there could be a very low range of popular contents to request. To justify this with simulations, depicted in Fig. 4.12, we fixed certain system parameters: $|\mathcal{N}_1|$=20, $|\mathcal{N}_2|$=100, ξ_1=200 GB, ξ_2=10 GB, σ_1=10, σ_2=2, and spanned the library size $|\mathcal{M}|$=500 to 5000.

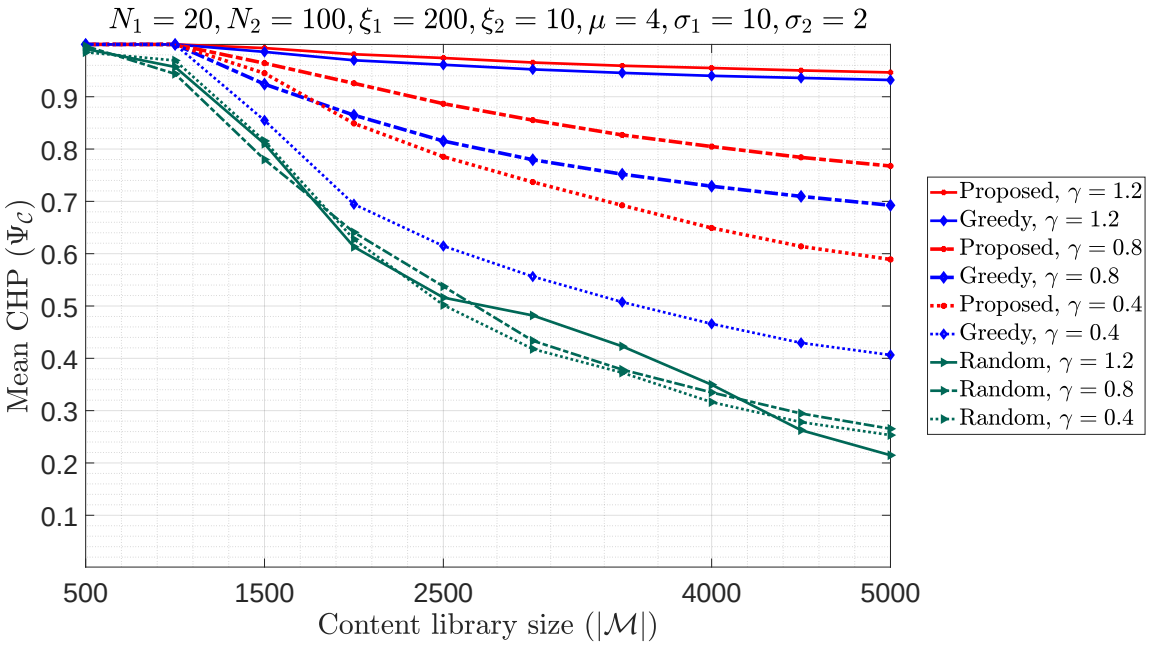

FIGURE 4.12: The CHP outcome for an increasing number of contents in the library and different Zipf parameters.

From the respective plot, we observe that an increase $\mathcal{M}$=500 to 5000 declines the CHP performance; intuitively, this also shows the MEC-computational efficiency. This relationship implies that caching in dense and modern cities is more challenging than in underdeveloped regions. Another important observation is that when γ is higher, the performance of the Greedy strategy outperforms nearly equal to the proposed (exact) one. This trend can be exploited to save computations at the MEC cluster head while making content placement for large γ values (e.g., γ>1.2). Nonetheless, when the popularity is spread across more videos in $\mathcal{M}$ (such as γ<0.8), the CHP performance of all strategies drops rapidly as the result of two difficulties: i) there are no outliers of viral contents and, ii) a huge number of contents are requested in the cache-limited network, which highlights the importance of employing sophisticated planning of the cluster-based content placement. As compared to the baseline strategies, the proposed strategy is shown to be highly robust against an increase $|\mathcal{M}|$ as well as to very high popularity heterogeneity (lower γ). Justifiably, the gap between the proposed and the Greedy increases with the $|\mathcal{M}|$

4.6.6 Analyzing computation costs and overheads

In this section, we evaluate the cost-wise performance of the proposed strategy in terms of the computation time necessary to complete the content placement. Recall that this operation shall be deployed on an epoch-by-epoch basis, where each epoch is expected to last for at least a few hours or days. For our evaluations, we used a PC equipped with the spec: Intel(R) Core i7-8550U CPU, 3.9 GHz, 20 GB RAM DDR3, 64-bit Windows 10 operating system. All strategies were implemented using a 64-bit version of Matlab R2018b, and the respective computation time measurements were averaged over multiple samples (more than 100 iterations per tick). Most of the computation results have been derived in line with the system parameter values used to derive the CHP measurement results of the previous subsections.

Impact of library size and number of MHs on computation

System parameters similar to the case in Fig. 4.12, the library size impacts computation time. As seen in Fig. 4.13, though $|\mathcal{M}|$ is increased from 500 to 5000, the computation time is in the range of sub-seconds, and it increases proportionally to $|\mathcal{M}|$ for all strategies under the scope.

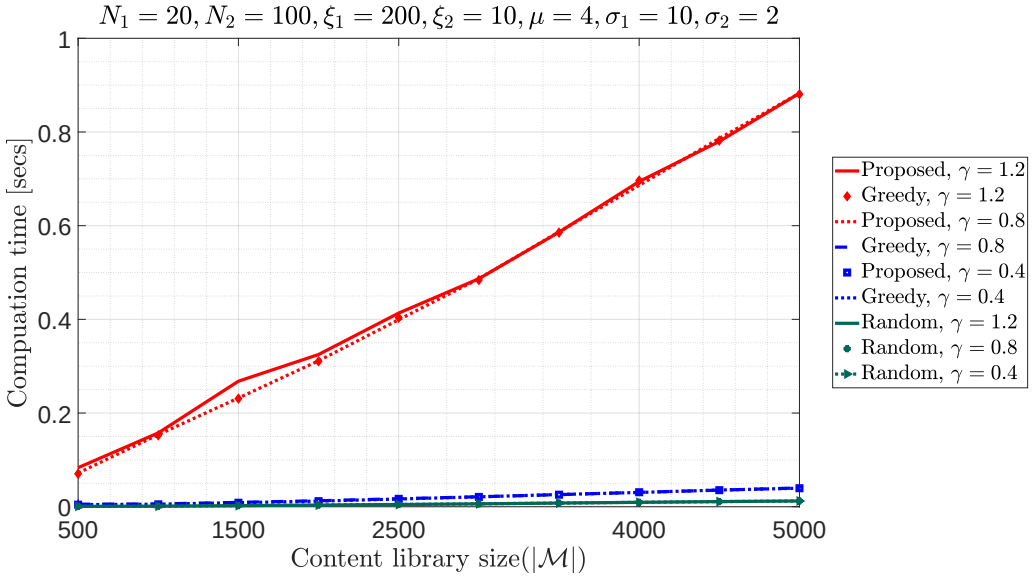

FIGURE 4.13: Computation time for increasing $|\mathcal{M}|$ under γ.

The computational performance of the BB-ZoMKP is dependent on the number of all types of MHs. As seen in Fig. 4.14, higher $|\mathcal{N}_2|$ costs higher computation time because it increases both the cluster cache size (mainly seen due to increasing $\mathcal{N}_1$) and the number of iterations per each helper edge (observed by increasing $\mathcal{N}_2$). Apart from that, the computation cost of the proposed strategy remains in a sub-seconds margin even when the number of SBS reaches $|\mathcal{N}_1|$=20 and the number of FBS is set to $|\mathcal{N}_2|$=100. This setup indicates that we can deploy the BB-ZoMKP strategy in the medium to large-sized MEC clusters.

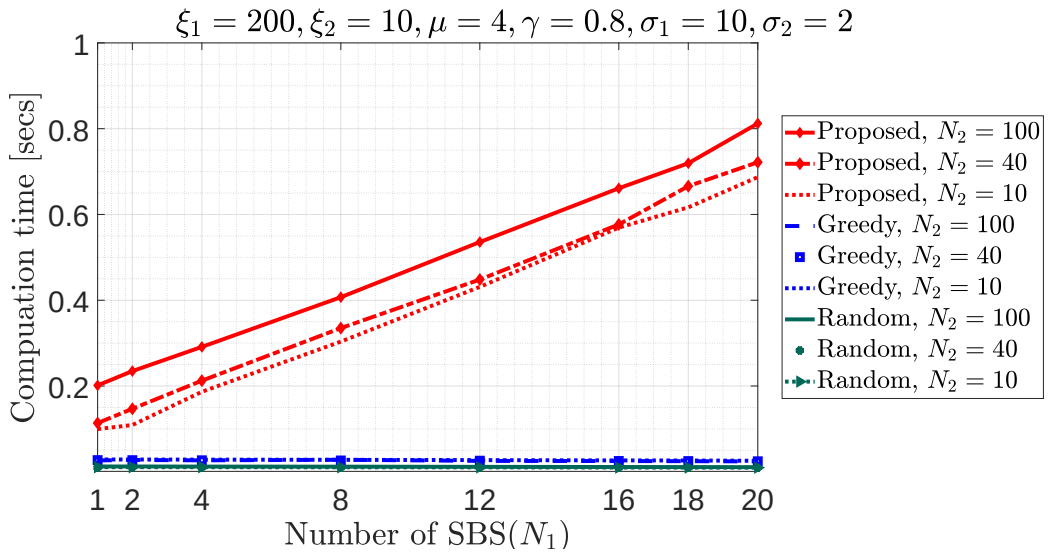

FIGURE 4.14: The computation time for an increasing number of SBS under different numbers of FBS.

Regardless of the number of caching edges, in both $\mathcal{N}_1$ and $\mathcal{N}_2$, the performance of the Greedy and Random very low and unaffected because, in the Greedy strategy, the main processing overhead comes from only sorting contents. In contrast, the Random strategy makes an arbitrary selection. This indicates the potential deployment of the Greedy strategy when it closely performs with the proposed strategy, such as when content popularity is highly skewed, or a large cache size is available.

Impact of MHs' cache size on computation

Before deploying the caching strategy, it is imperative to consider the available content size compared to the library size. For example, as shown in Fig. 4.15, the computation expense is very high when there is no sufficient cache size. Using ratio parameter β such that $\xi_1 = \beta \cdot \xi_2$, to increase the SBS mean cache size ξ_1 in proportion to the ξ_2, the MEC cluster consists of only FBS MHs for the case of β=0. At this point, there is a very low cache available, and it takes a huge amount of time to select optimal subsets from a large $\mathcal{M}$. When $\xi_1 \simeq \mu$ or $\xi_2 \simeq \mu$, the computation time exceeds several seconds because of the increased number of branches and bounds.

The other interesting design perspective is that when β=1, *i.e.*, the SBS and FBS mean cache sizes are similar or less heterogeneous, the computation time is high. This high computation is because the search state space is dominated by a vast volume of solutions with similar CHP performance. That involves the placement of only a few contents into the caches of similar-sized MHs, which means extensive iterations have to be made to eliminate sub-optimal solutions. This computation effect is alleviated when the available cache size ξ_2 is larger, enabling an enhanced content placement diversity per MH. However, the computation cost of the Greedy and Random strategies drops when the available cache size in the MEC cluster is high in both types of MHs.

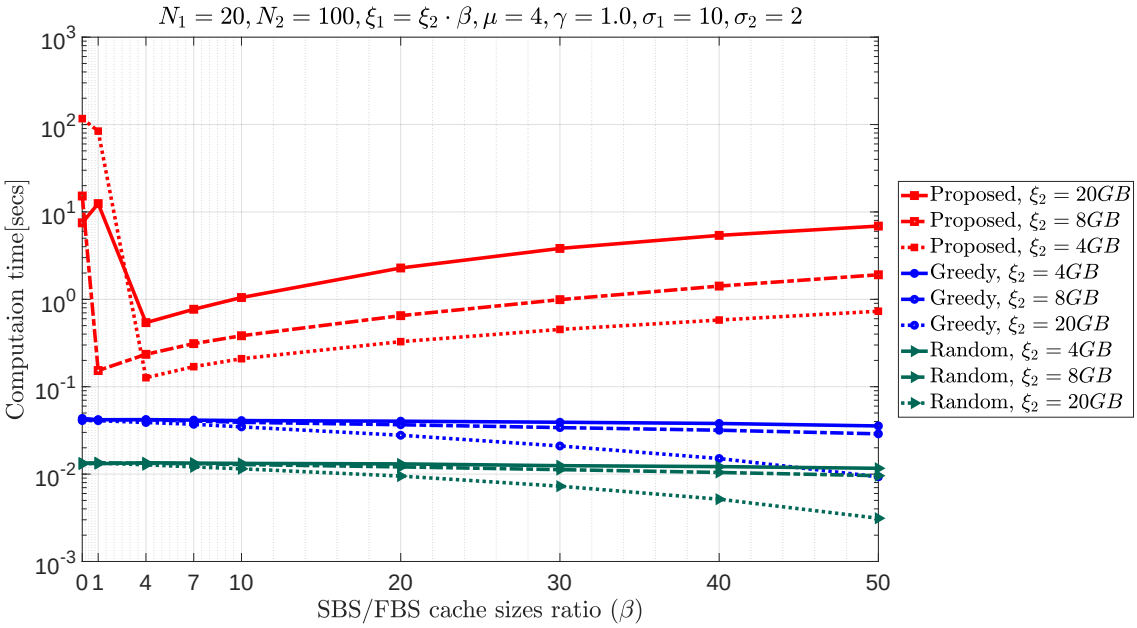

FIGURE 4.15: Computation time for increasing SBS/FBS rate under different FBS cache size values.

Impact of Zipf parameter and mean content size on computation

As the caching process is popularity-based, the computation time depends on the popularity distribution under certain cache and content size conditions. The plots in Fig. 4.16 show the mutual impact of the γ and μ values on the computation time under an increasing ξ_1. The computational complexity of the Greedy and Random strategies is roughly unaffected by γ and μ values.

The performance of the proposed content placement strategy above a certain ξ_1 value (such as double of ξ_2) and above, the computation time remains roughly unaffected by the γ and the μ. Similar to what we have observed in Fig. 4.15, this result follows from the fact that above a certain sufficient cache space, ξ_1 value, the content placement state-space includes solution branches exhibiting a diverse CHP performance and easily eliminates sub-optimal branches without going deeper roots. Accordingly, as shown in a similar figure, the impact of μ and γ becomes negligible for the proposed strategy, but its computation time scales up by the mean cache size ξ_2.

From the deployment perspective of the proposed BB-ZoMKP, it is wise to note that less heterogeneous MH cache sizes are computationally expensive. Similar to our findings from Fig. 4.15, when the mean SBS cache size value ξ_1 is comparable with the FBS mean cache size value of ξ_2 (which is set to 10 GB in this setup), an additional computation overhead is observed for the proposed strategy, in Fig. 4.16. A resembling overhead happens when ξ_1=0, which means there are only likely equal size FBS. The other observation is that, given γ=1.2, the computation time for μ=4 GB drops at ξ_1=20 GB and increases then after while it drops at ξ_1=60 GB for μ_1=8 GB, and increases equally with other cases. Both cases imply that when there is no sufficient cache size,

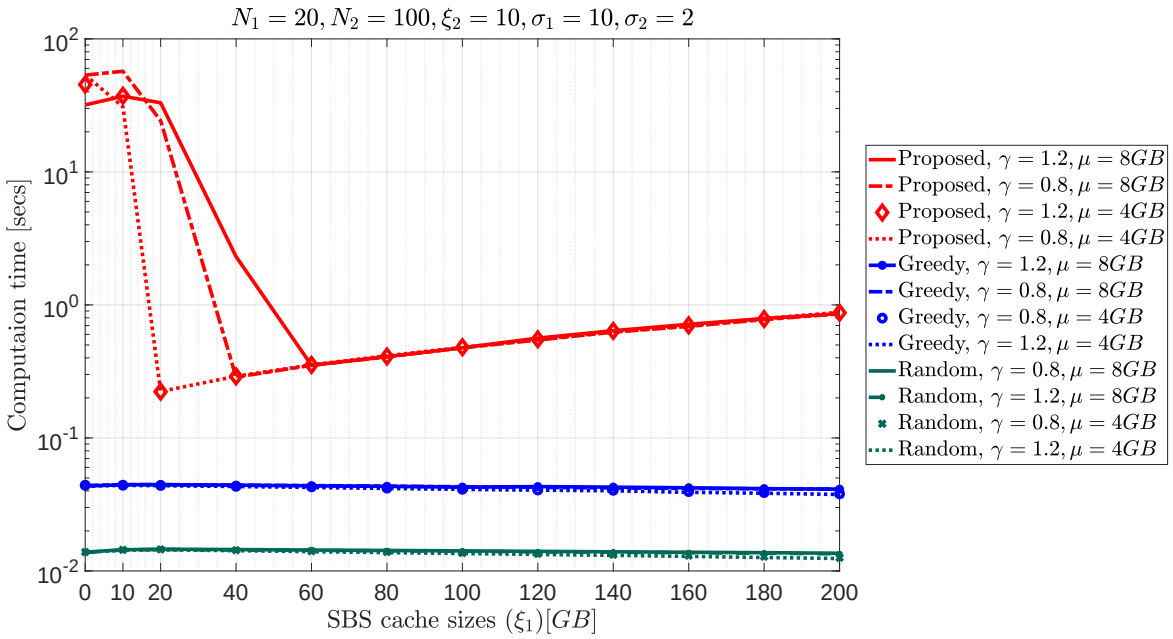

FIGURE 4.16: Computation time for increasing SBS mean cache size under two Zipf parameters and content sizes.

the iteration phase takes a long time because several instance branches are highly diverse. But after sufficient cache size, the computation complexity increases with the cache size due to the DP algorithm, *i.e.*, $O(|\mathcal{M}| \cdot L_n)$) complexity.

In some cases, the computation complexity also depends on popularity skewness γ such as when ξ_1 is small-sized. As shown in Fig. 4.16, given μ=8 GB, the computation drops lowest at γ=0.8 rather than γ=1.2 because of the heavy tail of the popularity distribution leading to large state-space. However, the Zipf parameter does not impact the computation time for small content mean size, μ=4 GB. This performance trend follows from the fact that a higher μ reduces the average number of contents cached per MH. For γ=1.2, it enforces the proposed strategy to explore solution branches with similar CHP performance in-depth, *i.e.*, branches that involve the placement of video contents with low popularity. Also, note that a very low computation time is required in scenarios where the proposed strategy exhibits superior performance.

4.7 Chapter Summary

In this chapter, we proposed a novel content placement architecture in which cache-enabled mobile helpers are grouped into MEC-enabled clusters and perform cooperative content placement, given a library of popular videos with defined size and popularity distribution. The caching process is modeled using the 0/1-MKP formulation, subjected to a set of cache constraints. To solve the formulation, we presented an exact placement strategy that employs the bound-and-bound enumeration, which discards solution branches that lead to sub-optimal performance.

The proposed bound-and-bound strategy exploits upper and lower performance bounds to eliminate sub-optimal branches without making deeper evaluations. Each iteration of the branches is optimally solved using the DP algorithm, which gives a pseudo-polynomial computational complexity. Here, the main objective is to maximize the placement of very popular contents into a cluster of several MHs. In doing so, the contents are placed at only one 'best' MH such that the combination instance brings the optimal solution. Once contents are efficiently placed at a hosting MH, thanks to its cooperation with the serving MH, it becomes straightforward to deliver them to requesting users at minimized service cost.

Using extensive system-level simulations, we have shown that the employment of the proposed bound-and-bound exact strategy is both efficient and computationally feasible in the context of content placement in realistic MEC-enabled HCN. Several design guidelines and best practices have been recommended through the simulation analysis, highlighting key performance trade-offs for MEC cluster formation and efficient content placement. The other important aspect of the proposed strategy is that the selection algorithm is scalable to any level of the HCN topology for predefined system parameters, such as content popularity, using any dynamic technique. The main drawback of this system model is that each content has equal popularity value among all MHs, which suppresses the individual interest of each MH in content. This limitation is solved using a novel formulation in the following chapter.

Chapter 5

Demand-aware Content Caching in MEC-enabled Cellular Networks

Because billions of smart devices are releasing an overwhelming amount of traffic content, the performance of the HCN is deteriorating. Another critical cause for content proliferation is the surge in data consumption per subscriber. As a mitigation mechanism, several caching strategies are proposed. Therefore, we need novel cache optimization techniques for the case where the demand of one content differs at each MH. Quite different from the state-of-the-art, these caching schemes must maintain the individual popularity of contents towards each MH. When we refer to 'demand-aware' caching, it is a tactical way to place contents at a caching edge that needs the content to the highest level. That specific placement instant ultimately contributes to maximizing an objective function. Generally speaking, 'demand' refers to the cumulative impact of both the intra-MH and the inter-MH popularity of contents.

In three sections, this chapter explains our cooperative *demand-aware* content caching strategies. The first section (5.1) covers a demand-aware caching strategy that does not allow content partitioning, while ensuring the individual interest of MHs towards each content. The second section (5.2) explains demand-aware caching, where contents can be systematically partitioned. The last section (5.3) covers a joint caching strategy that optimizes the end-to-end content placement and delivery phases.

5.1 Demand-aware Caching without Content Partition

This section dedicates to the scenario where demand-aware content caching is proposed under non-partitioning conditions. The placement problem is modeled using Separable Assignment Problems (SAP), where only one copy of the content is cached in a cluster of heterogeneous MHs. This strategy is a substantial extension of the BB-ZoMKP strategy, explained in Chapter 4, for being demand-aware, so it makes content delivery a trivially easy process.

5.1.1 Introduction

The ever-increasing content proliferation in the HCN continues to pose unavoidable challenges in meeting user expectations for seamless connectivity, high transfer rate,

and ultrafast responses. Still, powerful content caching strategies are needed to offload backhaul congestion, reduce service latency, and minimize utility costs in emerging networks [131], [139]. This effect is becoming a reality because powerful MEC computation is distributed in the RAN. When the MEC is integrated with the caching system, it reduces the completion latency and eases data analytics of the radio resource [140].

The performance of content caching strongly depends on its modeling tool and the effectiveness of the solution deployed, as seen in recent works. For example, authors in [34] use Multiple-choice Knapsack Problem to model cooperative content caching, while the authors in [30] model a cooperative caching strategy using a reward maximization problem. Authors in [141] propose cooperative caching using the Multiple Knapsack Problem and solve it using a bound-and-bound algorithm to maximize the cache hit probability in a cluster of heterogeneous MHs. However, these caching strategies are often modeled by misrepresenting system parameters, *e,g.*, content popularity, and solved mainly approximation algorithms and heuristics.

Though content caching is well-investigated, nearly all popularity-based strategies assume global popularity of contents, *i.e.*, by *aggregating* all requests received through all caching edges. It directly means that contents have similar popularity towards all MHs, which contradicts the real network behavior. This assumption underestimates the individual interests of MHs towards each content. This conjecture implies that the modeling tools used in the plethora of cache problem formulations misrepresent the characteristics of the real network. Therefore, contents are forced to be placed at MHs where they are not most 'demanded', increasing the intra-cluster service cost.

To address the above-mentioned critical issues, we develop a demand-aware caching strategy where content popularity varies at each MH. This novel work answers the following interlaced questions: i) Which content should we prioritize while caching at an MH? ii) At which MH should a video be cached to maximize its impact? These real problems are jointly modeled by the SAP [142], after making mathematically cumulating the analytics of the questions. The problem is solved optimally using an iterative algorithm. In this algorithm, contents are placed at the MH where they are most 'demanded' and maximize the *cache hit ratio* (CHR)– an objective function.

5.1.2 System model design

For this caching system, we focus on the downlink of a three-tier HCN, represented in Fig. 5.1 where several MHs prefetch popular contents from a huge content library in the first tier. As can be seen in the figure, the first *tier* refers to a set of network edges with the same content management that includes an MBS. Any MBS in this tier directly connects with the CS on the far Internet, using unlimited capacity links. It serves as a central head to the MEC-enabled cluster and has higher computation and content caching capability. The MBS gets frequently requested videos from the service providers and forms a huge library $\mathcal{M}$, as defined in Section 4.3. The second tier refers to a set $\mathcal{N}$ of multiple caching edges such as SBS, FBS, and smart UEs that are connected to an MBS using a capacity-limited fronthaul. These edges relay contents from the MBS to users, so they are the MH. Like the MBS, edges at the second tier have caching and

MEC-processing capabilities but are resource constrained. Each n^{th} MH has a set $\mathcal{U}_n$ of active users associated to it (n=1, 2, ..., $|\mathcal{N}|$). The third tier includes many UEs that request contents from their associated MH but can not cache or relay.

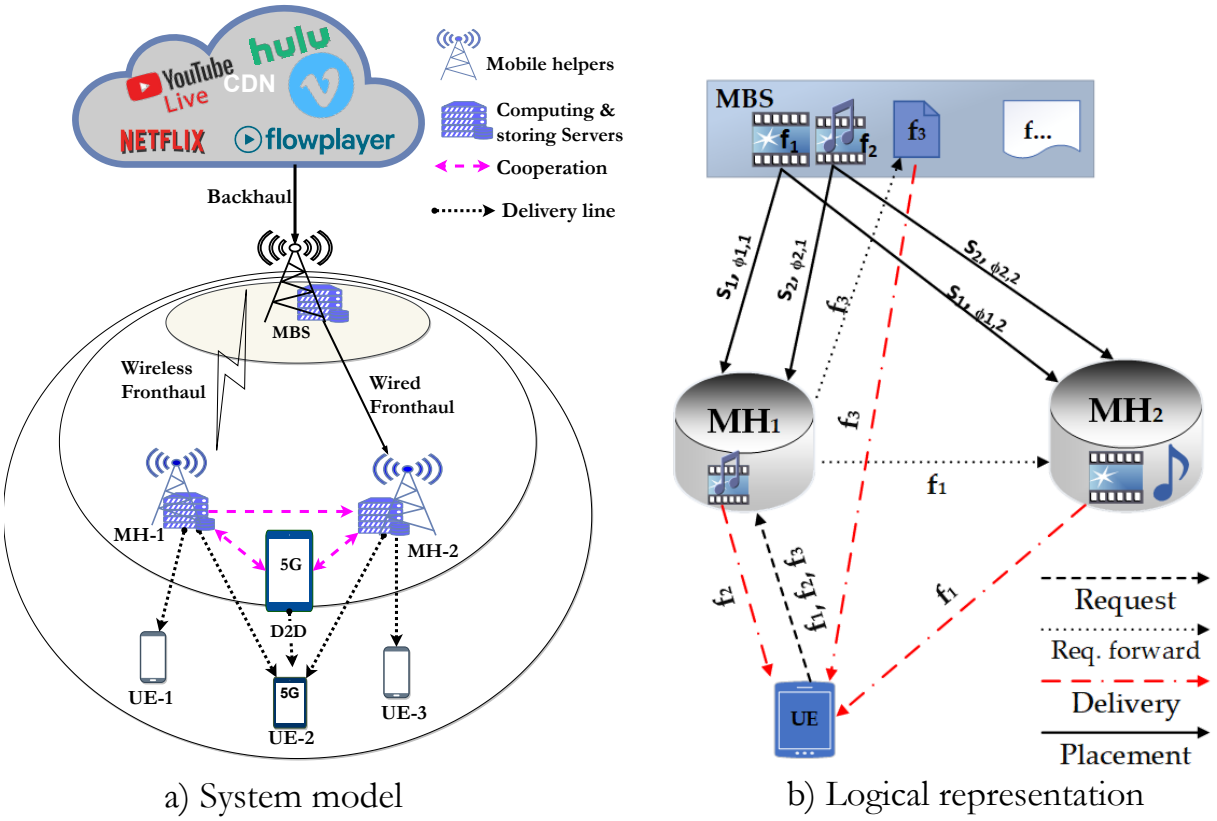

FIGURE 5.1: Cooperative demand-aware caching system.

We assume that an MBS has enough storage to contain a large library of popular video contents, $\mathcal{M}$=$\{f_m : 1 \leq m \leq |\mathcal{M}|\}$. Here, f_m is a unique content identifier, *e.g.*, Uniform Resource Identifier (URI), or Uniform Resource Locator(URL). Each content's *popularity* ($\rho_{m,n}$), which shows the popularity of content f_m at n^{th} edge, is different towards each MH. More importantly, the popular contents have different *request probability* ($\phi_{m,n}$) across the MHs (both parameters described in subsection 5.1.4). This difference means that f_m is highly demanded at one MH but not important at another. However, the content size (s_m) does not differ and is stored in a size vector $\mathcal{S}$=$\{s_1, s_2, \cdots, s_{|\mathcal{M}|}\}$.

5.1.3 System MEC computation

For scaling convenience, we only focus on optimizing content caching in a single cluster where a subset ($\mathcal{C}_n$) of contents is to be selected for each MH as a result of some computing tasks. Here, the caching system is a bidirectional MEC process where contents are prefetched from a single source (mainly the MBS but also the CS) to multiple destinations (heterogeneous MHs). At the same time, *services* and *outputs* are uploaded

to central MBS. The computing processes, whose flow diagram is detailed in Fig. 5.2, are done in every epoch but independent of the transmitting tasks to another tier.

The multi-task MEC computation starts by collecting the RAN data from all UEs, including the request rate that f_m receives through n^{th} MH and edge's temporal-spatial information (*info*) such as the location of UEs (MH process ① - ④). At a fractional time interval of the epoch, the MH updates its *request profile* ($\mathcal{R}_n$). At least once before the epoch time ends, the MH calculates its local popularity ($\mathcal{P}_n$). Then, the MH transmits these outputs to the first tier and uploads them to the central MBS. Within a new epoch, the MBS updates its request matrix ($\mathcal{R}$); then after, it computes the spatial request ratio ($\mathcal{H}$) and request probability (Φ) of the cluster. After the data analytics, the central head applies the proposed content selection strategy (see details in Subsection 5.1.8). While doing so, it centrally controls the cache decision (MBS compute process ❶-❺). As a result, the MBS transmits the selected subset of contents ($\mathcal{C}_n$) and its cache-decision lookup table ($\mathbf{x}$) to respective MHs (detailed in Subsection 5.1.5).

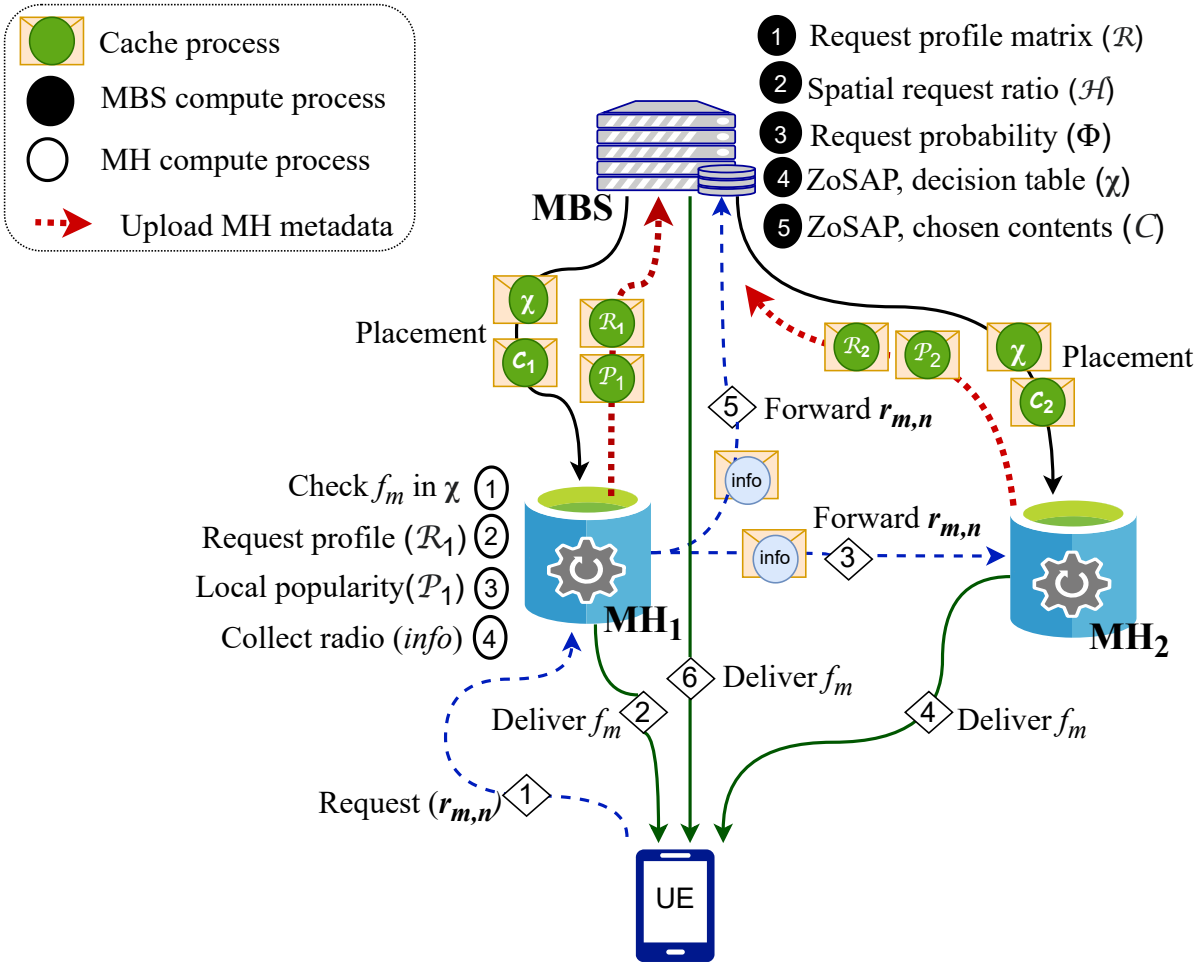

FIGURE 5.2: MEC computing processes and cooperative cluster caching.

5.1.4 MEC radio information analytics

Recall that every MEC-enabled MH stores content request profile can perform data analytic tasks on the radio information, and transmits computed results to the MBS. Independently, the MBS collects all request rates $\mathcal{R}$=$\{r_{m,n} : \forall f_m \in \mathcal{M}, \forall n \in \mathcal{N}\}$ and computes the last two of the following three important parameters.

Local popularity

It refers to the probability of contents that associated UEs request, compared to the set of contents in an edge. The local popularity ($\rho_{m,n}$) helps to answer the question: which popular content should be prioritized while caching at a specific MH? By ranking the contents in descending order of their local request profile, contents popularity is modeled using the Zipf distribution. Let m_n denote the rank of content in the request rate vector of n^{th} MH, then the edge forms its local popularity vector (LPV) $\{\rho_{m,n}\}$ after computing file popularity, by redefining Eq. 2.1, as:

$$\rho_{m,n} = \frac{m_n^{-\gamma}}{\sum_{j=1}^{|\mathcal{M}|} j_n^{-\gamma}}, \forall n \in \mathcal{N} \tag{5.1}$$

where $\gamma \geq 0$ is the Zipf parameter of the popularity distribution respective to each MH. Likewise, in the ZoMKP case, the γ=0 indicates that all contents have equal popularity in an MH and follow a uniform distribution. In contrast, higher values such as γ=2 show that only a few contents are highly popular in that particular MH. For computational convenience, γ value is assumed to be similar in all MHs and $\sum_{m=1}^{|\mathcal{M}|} \rho_{m,n}$=1, $\forall n \in \mathcal{N}$. The MBS forms a popularity matrix $\mathcal{P}$=$\{\rho_{m,n} : m = 1, ..., |\mathcal{M}|, \forall f_m \in \mathcal{M}, \forall n \in \mathcal{N}\}$, which is an equivalent matrix to a vector defined in Section 4.3.

Spatial request ratio

This metric refers to the spatial demand of content across a series of MHs in the MEC cluster. It is the relative demand that determines: at which MH is the content most demanded? Let $r_{m,n}$ denote the total number of requests that f_m received from all active users through their common local MH (n). Its spatial request ratio ($\eta_{m,n}$) at n is estimated by the central MBS, as:

$$\eta_{m,n} = \frac{r_{m,n}}{\sum_{n=1}^{|\mathcal{N}|} r_{m,n}}, \forall f_m \in \mathcal{M} \tag{5.2}$$

where it is clear that $\sum_{n=1}^{|\mathcal{N}|} \eta_{m,n}$=1, $\forall f_m \in \mathcal{M}$. Then, it establishes the request ratio matrix $\mathcal{H}$ such that $\mathcal{H}$=$\{\eta_{m,n} : \forall n \in \mathcal{N}, f_m \in \mathcal{M}\}$.

Request probability

The request probability ($\phi_{m,n}$) is the linear cumulative impact of the above two factors on a content, *i.e.*, spatial request rate across MHs, and relative demand of contents within each MH. It shows the real demand level of video content that a specific helper n and the access probability in a system. The $\phi_{m,n}$ value of content f_m at serving helper n is estimated as:

$$\phi_{m,n} = \rho_{m,n} \cdot \eta_{m,n} \tag{5.3}$$

and a matrix of request probabilities (Φ), Φ=$\{\phi_{m,n} : \forall f_m \in \mathcal{M}, \forall n \in \mathcal{N}\}$ is formed to be used by the demand-aware content selection algorithms for the placement.

5.1.5 Content placement phase

Cache-enabled clusters of $|\mathcal{N}|$ cooperative MHs, that are governed by one MBS, are formed at the second tier. The MHs are dynamically admitted to each cluster using any convenient policy. The MHs are meshed through a broadband connection, either wired or wireless, but not all are necessarily interconnected. All MHs bounded in a cluster cooperatively cache subsets of popular video contents ($\mathcal{C}_n$), from the MBS, during off-peak periods. Here, we focus on cache optimization in a single cluster, within which a subset of contents is placed at each MH until its cache $Ln[bits]$ is full, and it serves a maximum of $|\mathcal{U}_n|$ active UEs at a time.

After making the cache decision, based on results from data analytics, the MBS places a subset of contents $\mathcal{C}_n$ at n^{th} MH, where $\mathcal{C}_n$=$\{f_m : 1 \leq m \leq |\mathcal{C}_n|, f_m \in \mathcal{M}\}$. The superset of cached contents in the cluster is $\mathcal{C}$, $\mathcal{C}$=$\bigcup_{n=1}^{|\mathcal{N}|} \mathcal{C}_n$, where $\mathcal{C}_n \subseteq \mathcal{C} \subseteq \mathcal{M}$. Again worth noting that during caching, we consider: i) contents are not partitioned, *i.e.*, either an entire part is cached or not selected at all, ii) there is no content overlap across all MHs in a cluster, *i.e.*, $\mathcal{C}_n \cap \mathcal{C}_k$=$\emptyset$, $n \neq k$, $k \in \mathcal{N}$, iii) the popularity $\rho_{m,n}$ and demand $\phi_{m,n}$ of content differs towards each MH, *i.e.*, $\rho_{m,n} \neq \rho_{m,k}$ and $\phi_{m,n} \neq \phi_{m,k}, n \neq k, \forall n, k \in \mathcal{N}$. Meanwhile, whether a content is found in the cache of an MH is indicated by the cache-decision parameter $x_{m,n}$ such that the MBS passes the matrix ($\mathbf{x}$) to all MHs, $\mathbf{x} = \{x_{m,n} : \forall f_m \in \mathcal{M}, \forall n \in \mathcal{N}\}$. This is an intra-cluster cache lookup table to enhance the cooperation among the MHs during content delivery (detailed in Subsection 5.1.6) to requesting UEs.

Meanwhile, the availability of video contents in the cluster is measured by *cache hit ratio* (Φ), which is the ratio of 'the requests of contents having successful cache hit event' and the 'total received requests by all contents' [81], [108]. Given $|\mathcal{U}_n|$ active users and request probability ($\phi_{m,n}$) of contents, the weighted number of requests towards f_m is estimated by: $r_{m,n}$=$|\mathcal{U}_n| * \phi_{m,n}$ [30]. Having this, the CHR in the cluster, defined over $\mathbf{x}$, is estimated as:

$$\Phi(\mathbf{x}) = \frac{\sum_{n=1}^{|\mathcal{N}|} \sum_{m=1}^{|\mathcal{M}|} x_{m,n} \cdot r_{m,n}}{\sum_{n=1}^{|\mathcal{N}|} \sum_{m=1}^{|\mathcal{M}|} r_{m,n}} \tag{5.4}$$

5.1.6 Content delivery phase

Once the contents are placed on the network edge, they are served to users whenever lively requested. Let us look at the logical diagram in Fig. 5.2, which displays the data analytics, MEC computations, and bidirectional caching protocol. Recall that, from their cooperative service perspective, there are three functionality types of MHs. The first one is a *local* MH, with which the UE is associated. The second type is the *hosting* MH, which contains the requested content. The third type is *serving* MH, which ultimately delivers the content to the requesting UE. But mind that the MHs might have multiple roles, *e.g.*, a local MH might be the serving and hosting MH.

We assume that every UE, falling within the MEC-cluster coverage, issues request $r_{m,n}$ for a content f_m only through its local MH (n). Then, the local MH checks the whereabouts of f_m from its lookup table ($\mathbf{x}$). As a result, one of the following three events happens: i) a *local cache hit* happens ($x_{m,n}$=1), meaning that f_m is found at n and immediately served to UE, ii) a *local cache miss* happens, meaning that f_m is not found in n but at any other hosting MH k ($x_{m,k}$=1, $n \neq k$). In this case, the local MH forwards $r_{m,n}$ and radio information to k so that the hosting MH sends the content to the serving MH, iii) a *cache miss* happens ($x_{m,n}$=0, $\forall n \in \mathcal{N}$), meaning that f_m is not found in any of MHs in the cluster. So, the local MH forwards the $r_{m,n}$ and information to the central MBS, which is supposed to have the content in its $\mathcal{M}$.

In the event of a cache miss, the content delivery is not subjected to any optimization sphere so that the requested f_m is directly served to UE via the established downlink. In addition, its request history is registered for future data analytics by the appropriate network edge. Intermediately, if the content is required to be placed in the cluster, any convenient cache refreshment techniques can be used [3]. Due to handover overheads, the content delivery link is still limited to only via the serving MH.

5.1.7 Problem formulation

In this section, we focus on cluster-wise optimization of content placement of (detailed in Subsection 5.1.5). Hereof, the objective function of the MEC system is to maximize the availability of popular contents, in terms of the CHR, at the network edges but without violating their cache limit. To maximize content availability, we aim at selecting $|\mathcal{N}|$ number of non-overlapping subsets ($\mathcal{C}_n$) of highly popular contents, that will be cached at appropriate MH. As a result, the objective function is globally optimized.

Subjected to some constraints, the maximization problem is given as:

$$P_{5.1}: \max_{\forall x_{m,n} \in \mathbf{x}} \Phi(\mathbf{x}) \tag{5.5a}$$

$$\text{subject to:} \quad \sum_{m=1}^{|\mathcal{M}|} x_{m,n} \cdot s_m \leq L_n, 1 \leq n \leq |\mathcal{N}| \tag{5.5b}$$

$$\sum_{n=1}^{|\mathcal{N}|} x_{m,n} \leq 1, 1 \leq m \leq |\mathcal{M}| \tag{5.5c}$$

$$\sum_{m=1}^{|\mathcal{M}|} x_{m,n} \cdot \rho_{m,n} \leq 1, 1 \leq n \leq |\mathcal{N}| \tag{5.5d}$$

$$\sum_{n=1}^{|\mathcal{N}|} x_{m,n} \cdot \eta_{m,n} = 1, 1 \leq m \leq |\mathcal{M}| \tag{5.5e}$$

$$x_{m,n} \in \{0,1\}, 1 \leq n \leq |\mathcal{N}|, 1 \leq m \leq |\mathcal{M}| \tag{5.5f}$$

Here, constraint (5.5b) indicates that the sum of sizes of all contents in each MH should not exceed its cache size and (5.5c) limits that content is cached at only one MH. The constraint in (5.5d) indicates that the popularity of cached contents at each MH does not exceed 1 and it may have different distribution among MHs. Constraint (5.5e) shows the normalized demand rate and cached content should be requested by at least one MH. Lastly, the constraint in Eq. 5.5f limits that contents should not be partitioned during the placement process.

In the formulated problem (5.5), the objective function $\Phi(\mathbf{x})$ is globally maximized over an extremely huge number of combinatorial subsets, from which the final caching decision table $\mathbf{x}$ is chosen. The most important and realistic aspect of this content caching formulation, with regard to cellular network behaviors, is that both $\rho_{m,n}$ and $\eta_{m,n}$, collectively $\phi_{m,n}$, vary for different MHs. Using the $\phi_{m,n}$, to capture both impacts, we modeled it using the 0/1-*Separable Assignment Problem* (ZoSAP).

5.1.8 Proposed SAP-based solution

Problem $\mathbf{P}_{5.1}$ in Eq. 5.5a is an NP-hard and extremely difficult to solve optimally, in this form; meaning, to get an optimal content placement decision $x_{m,n}$. Instead, we reduced the problem into $|\mathcal{N}|$ subproblems and solve them with an iterative approach. Each subproblem is modeled by the ZoSKP formulation (Subsection 4.5.1) and solved using the DP-ZoSKP strategy at a pseudo-polynomial time of $O(|\mathcal{M}| \cdot L_n)$. Then, an iterative filtering mechanism is superimposed on the solutions of each subproblem.

In the proposed cache content selection procedure, represented in Algorithm 6, we first run the *Initialize* function where variables are defined, such as cache-decision ($x_{m,n}$) and subset of contents in the cluster ($\mathcal{C}$). Then, using *Availability* function (lines 5 to 11), we calculate the available free cache sizes from all MHs in the cluster and yet unassigned contents from library $\mathcal{M}$.

Algorithm 6: Proposed D-ZoSAP caching strategy

Input: $\mathcal{M}, \mathcal{R}, \mathcal{S}, \mathcal{N}, \{L_n\}$
Result: $x_{m,n}$
1 [**Initialize**]
2 $x_{m,n} = zeros(|\mathcal{N}|, |\mathcal{M}|)$;
3 $\mathcal{C} = \emptyset$;
4 **Function:** `D-ZoSAP`$(\mathcal{R}, \mathcal{S}, \mathcal{M}, \{L_n\})$
5 [**Availability**]
6 **for** $n = 1$ *to* $|\mathcal{N}|$ **do**
7 $\bar{L} = \left\{\bar{L}_n : \bar{L}_n = L_n - \sum_{m=1}^{|\mathcal{C}_n|} s_m \cdot x_{m,n}\right\}$;
8 $\bar{\mathcal{M}} = \{f_m : x_{m,n} = 0, f_m \in \mathcal{M}, \forall n \in \mathcal{N}\}$;
9 $\bar{\mathcal{R}} = \left\{\phi_{m,n} : \phi_{m,n} \in \mathcal{R}, \forall f_m \in \bar{\mathcal{M}}\right\}$;
10 $\bar{\mathcal{S}} = \left\{s_m : s_m \in \mathcal{S}, \forall f_m \in \bar{\mathcal{M}}\right\}$;
11 **end**
12 [**Round**]
13 $\hat{x}_{m,n} = zeros(|\mathcal{N}|, |\mathcal{M}|)$;
14 **for** $n = 1$ *to* $|\mathcal{N}|, \forall m \in \mathcal{M}$ **do**
15 $\hat{x}_{m,n} =$ **DP-ZoSKP**$(\bar{\mathcal{M}}, \bar{\mathcal{R}}(\forall m, n), \bar{\mathcal{S}}, \bar{L}_n)$ (*apply Algorithm-2*);
16 **end**
17 **for** $m = 1$ *to* $|\mathcal{M}|$ **do**
18 **if** $(\sum_{n=1}^{|\mathcal{N}|} \hat{x}_{m,n} > 1)$ **then**
19 $\hat{x}_{m,n} = \begin{cases} 1, & \text{at} \quad n : \phi_{m,n} = \max_{n \in \mathcal{N}} \bar{\mathcal{R}}(m,n) \\ 0, & \text{else;} \end{cases}$
20 **end**
21 **end**
22 $x_{m,n} = x_{m,n} \oplus \hat{x}_{m,n}$;
23 Call **Availability**;
24 **for** $n = 1$ *to* $|\mathcal{N}|$ **do**
25 **if** $(\bar{L}_n \geq \min(s_m), \forall s_m \in \bar{\mathcal{S}})$ **then**
26 $\bar{L} = \bar{L} \cup \{\bar{L}_n\}$;
27 **end**
28 **end**
29 **repeat**
30 Go to **Round**;
31 **until** $(\bar{L}_n < \min(s_m), \forall \bar{L}_n \in \bar{L}, \forall s_m \in \bar{\mathcal{S}})$;
32 **return** $x_{m,n}$;
33 **end**

In the *Round* step (lines 14-16), every MH gets its candidate subset of contents $\mathcal{C}_n$ by the temporary decision matrix $\hat{x}_{m,n}$, using DP algorithm (as explained in Subsection 4.5.1 and Algorithm 2). After candidate subsets selection, if any content f_m exists in

multiple subsets, it is assigned to only the subset, *i.e.*, $\hat{x}_{m,k}$=0, where f_m has the highest $\phi_{m,n}$ value (lines 17-21). Therefore, they are removed from all other subsets, i.e., $\hat{x}_{m,k}$=0,$\forall k \neq n$. The decision matrix $x_{m,n}$ is updated by bitwise XOR with $\hat{x}_{m,n}$ (line 22). All resulting subsets so far are feasible but not optimal until all MHs can no longer accept contents. Rather, the expected value of rounded CHR is at least (1-(1-$\frac{1}{|\mathcal{N}|}$)$^{|\mathcal{N}|}$) times the optimal objective value (interested readers are referred to [142] for the proof).

Still, there could be available cache space and free contents after each $x_{m,n}$ update step, so *Availability* function is called, and only relevant spaces are considered (lines 23-28). Then, the content selection is repeated whenever there is space and unassigned content (line 30). The final caching decision of all iterations in an epoch is found when there is no space to cache any content. This strategy can also be used for cache *refilling* after a profile update– an intermediary step before full-scale caching.

The proposed content caching strategy for multiple heterogeneous MHs, decides whether any free f_m is placed at its most 'demanding' MH but without partitioning. We call this policy as *demand-aware zero/one separable assignment problem (D-ZoSAP)* caching strategy. It gives an optimal content selection at a pseudo-polynomial time complexity of $O(|\bar{\mathcal{M}}| \cdot \sum_{n=1}^{|\mathcal{N}|} \bar{L}_n)$ for each iteration. This time complexity vanishes very fast because the number of free contents and the size of space decay exponentially per iteration.

5.1.9 Numerical results and discussions

In this subsection, we evaluate the performance of the D-ZoSAP placement strategy by comparing it to the state-of-the-art that assumes equal content popularity for each MH. Thus for the two baseline strategies, the popularity of content does not change towards all MHs but their global popularity is estimated by taking the aggregated request rate from the cluster. These baseline strategies: i) *bound-and-bound zero/one-Multiple Knapsack Problem (BB-ZoMKP)* caching, an exact strategy that models the content caching by MKP and iteratively solves each iteration using DP (details in [141]), ii) *Greedy-MKP* caching that selects the most popular contents after sorting them in non-increasing order of their popularity, which is very often used [138]. It sequentially fills each MH using available free contents per iteration.

We evaluate the performance of the D-ZoSAP caching strategy in terms of CHR (see Eq. 5.4), which shows the percentage of content requests successfully served by the helpers compared with the total number of requested contents in the cluster. The CHR is the expectation of user request to be a cache hit event, so it is a more general performance measuring metric [143]. We have two CHR in kind: *one-hop* CHR (Φ) and *local* CHR ($\hat{\Phi}$). The one-hop CHR (OCHR) is a cluster-wise success ratio, including those served by the hosting MHs (*i.e.,* after cluster cache hit events). The local CHR (LCHR) is part of the OCHR but is specific to the direct success rate by the local MHs only (*i.e.,* after local cache hit events). An OCHR value close to the LCHR implies that UEs' interest is concentrated on associated MHs and provides remarkable latency-sensitive applications with guaranteed latency. It enhances content local availability and boosts data retrieval from the most convenient location. On the other hand, an

OCHR value much higher than the LCHR shows that the UEs' interest is spread to multi-hop edges, thus aggravating network traffic overhead during delivery [108].

We did extensive system-level MATLAB simulations (on 20 GB RAM, Intel i3-7100 CPU) for several network scenarios. We considered a 3-tier MEC-cluster which contains three types of computing and caching edges: one central MBS at the first tier, 20 SBS ($\mathcal{N}_1$), and 100 femto-base stations (FBS) ($\mathcal{N}_2$) at the second tier; *i.e.*, $|\mathcal{N}_1|+|\mathcal{N}_2|$=120 MHs. The cache size sets of the SBS and FBS, $\mathcal{L}_1$ and $\mathcal{L}_2$, follow a normal distribution with mean and variance pairs of (ξ_1, σ_1^2) and (ξ_2, σ_2^2), respectively. That is, $\mathcal{L}_1 \simeq norm(\xi_1, \sigma_1^2)$ and $\mathcal{L}_2 \simeq norm(\xi_2, \sigma_2^2)$, while $\mathcal{L}^{1x120} = \{L_n : n = 1, ..., |\mathcal{N}|\} = \mathcal{L}_1 \cup \mathcal{L}_2$. In simulation cases where these parameters have to be fixed, we take $\xi_2 = 20$ GB, $\sigma_1^2 = 10$, and σ_2^2 = 2, which give us the required network heterogeneity. The number of active UEs associated with each serving MH is represented by an exponential distribution: $\mathcal{U}_n \simeq exp(\delta)$, with mean value δ=10.

We consider a huge library with $|\mathcal{M}|$ = 5,000 contents, at the MBS, such that their size $\{s_m\}$ follow an exponential distribution of mean μ, *i.e.*, $\mathcal{S}^{1x5000} \simeq exp(\mu)$. Worth to note that $L \ll \sum_{m=1}^{|\mathcal{M}|} s_m$, where $L = \sum_{n=1}^{|\mathcal{N}|} L_n$. The local content popularity ($\rho_{m,n}$), with respect to each MH, is modeled by the Zipf distribution, where all demanded contents are ranked based on the number of requests they received in that MH at a given time epoch. For that, we fixed γ=1.0, which shows that 10% of contents account for 75% of the local popularity in our $|\mathcal{M}|$. On the other hand, the request ratio of a content ($\eta_{m,n}$) follows a Poisson distribution with rate parameter λ=100 requests per epoch. The spatial request probability ($\phi_{m,n}$) is the expectation of the product distribution of the two independent variables, *i.e.*, $\phi_{m,n}$=E[$\rho_{m,n}.\eta_{m,n}$].

Technically, we focus on analyzing large videos that greatly burden the network performance. These are long-lasting popular videos, such as the 4K types, whose content size is adapted according to average bit rate values in 'recommended upload encoding settings for YouTube' (by Google company, [123]) and 'guidelines for video delivery over a mobile network' (by NTT-DOCOMO company, [124]). Based on these guidelines and a survey on size specifications from a few contents delivery companies (look at Table 4.3), we fixed μ=4 GB, which corresponds to an HDR (4K) video of resolution 3840×2160 (2160p) at High Frame rate, the average length of 8 minutes, and bit rate of 66 Mbps (look at Table 4.2).

In the following subsections, we explain the impacts of system parameters on caching performance, using some of the extensive simulations, in a side-by-side plot: i) a CHR value (Φ) for the entire cluster and, ii) a *local mean* CHR value ($\hat{\Phi}$) for an MH. Recall that Φ indicates the availability of all requested contents in the cluster, regardless of the MH they are placed; maybe, not in serving MH but in hosting MH. A very high CHR value, let's say Φ=0.95, means that almost all requested contents are available in the MEC cluster (a high cache hit event happens). More specifically, the local ($\hat{\Phi}$) shows the availability of requested contents within each associated MH (which the UE makes the request through). This value is calculated by taking the mean CHR of all MHs so that, in contrast with the Φ, it shows the degree of demand-based content placement.

Impact of cache size on cache hit ratio

For this scenario, the ξ_1 value stepped from ξ_1=0 GB (means there no SBS) to ξ_1=500 GB, while ξ_2=20 GB for all FBSs. Thus, at every stage, the cluster cache size is estimated as: $L = |\mathcal{N}_1| \cdot norm(\xi_1, \sigma_1^2) + |\mathcal{N}_2| \cdot norm(20, \sigma_2^2)$ GB. We popularity distribution parameter γ=1.0 and contrasting content mean sizes μ=4 GB and μ=8 GB.

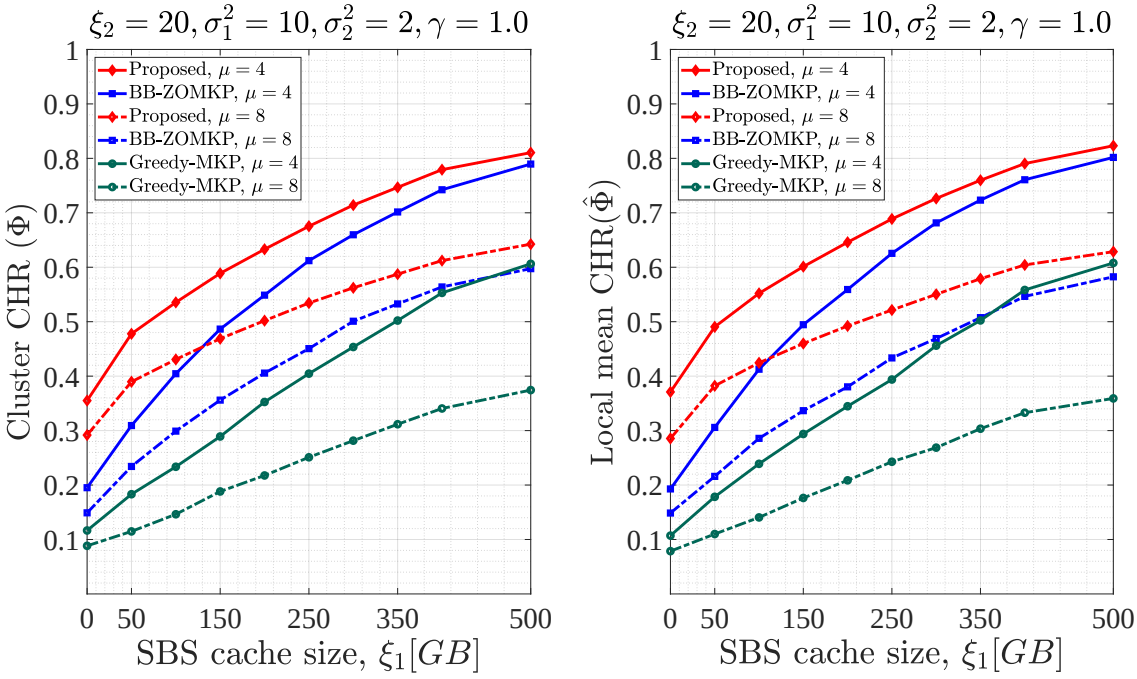

FIGURE 5.3: For increasing cache size: a) CHR for the cluster (left), b) Mean CHR for each helper (right).

The two plots in Fig. 5.3 indicate the CHR of a cluster (left) and the mean local CHR (right) per MH. As can be seen, the $\hat{\Phi}$ is as large as the Φ, indicating that almost all contents are placed at MHs through which they were highly requested. This proves that contents are placed based on their demand level. We also notice that, for all strategies, as the cache size increases, both cluster and local CHRs increase. This is because the cluster gets sufficient cache space to place more contents. However, as μ steps up from 4 GB to 8 GB, the performance gap between corresponding Φ outcomes linearly widens (a similar pattern is seen for $\hat{\Phi}$). At an equal level of CHR or content availability, doubling the μ value needs about 2.5$\times$ ξ_1 cache space. This indicates that only enlarging cache size does not bring the expected network performance but also depends on content sizes.

In all cases, the proposed D-ZoSAP caching strategy outperforms baseline strategies. It is extremely useful when we have smaller cache sizes; *e.g.*, at ξ_1=50 GB, it gives a 150% hit ratio than the BB-ZoMKP and 260% than the Greedy-MKP strategies, both locally and globally. This is achieved because the D-ZoSAP algorithm places every

content at the MH that demands it the most, while the baseline strategies do not handle this content spatial demand. The D-ZoSAP might not be required when the cluster cache size is very large. We separately proved that D-ZoSAP and BB-ZoMKP strategies equally perform for $L \geq 0.70 \cdot \sum_{m=1}^{|\mathcal{M}|} s_m$. Here, the BB-ZoMKP outperformed the Greedy-MKP strategy.

Impact of content size on cache hit ratio

The availability of contents after a caching process depends on the content size under some system parameters. The two plots in Fig. 5.4 give us more details on the relation of both Φ and $\hat{\Phi}$ with μ, under two popularity skewness γ=0.4 and γ=1.2. The video content mean size (μ) of the exponential distribution is increased μ=2 GB to μ=16 GB while the cache size of MHs is fixed with ξ_2=20 GB and ξ_1=20·ξ_2.

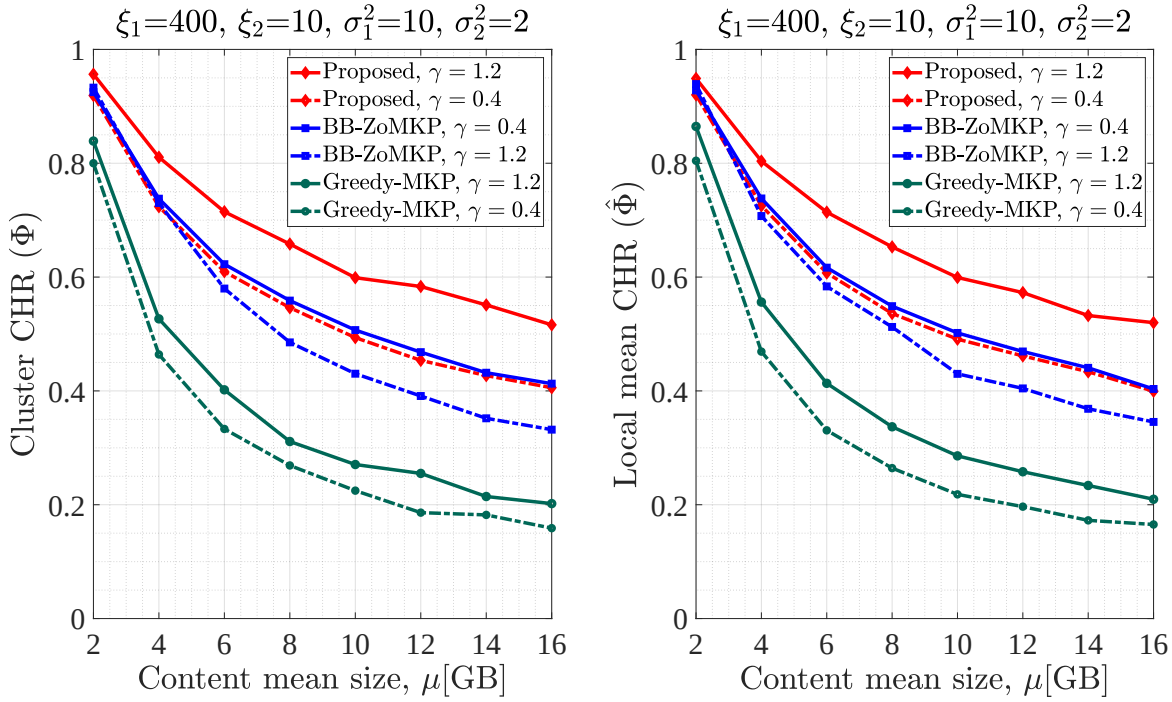

FIGURE 5.4: For increasing content size: a) CHR for the entire cluster (left), b) Mean CHR for each helper (right).

The straightforward observation is that Φ and $\hat{\Phi}$ show a similar pattern, so the CHR at each MH is nearly equal. For both cases, we observe that all strategies having small-sized (μ=2 GB) video contents give a very high hit ratio, *e.g.*, from Φ=0.85 (Greedy-MKP) to Φ=0.95 (D-ZoSAP). When content size gets bigger to μ=16 GB, this performance exponentially decays to Φ=0.15 and Φ=0.51, respectively. This is because the cache size becomes a severe constraint to caching larger contents.

In another comparison, the CHR value generally declines when the popularity index gets lower from γ=1.2 (highly skewed) to γ=0.4 (close to uniform). For example, when the proposed strategy is applied on nearly uniform popularity but large-sized

contents (*e.g.*, μ=16 GB), their availability decreases from 52% to 40% while the BB-ZoMKP drops from 41% to 33% because the content size constraint becomes tougher while the request probability of requested content decreases due to the dispersion of content popularity to a larger number.

In all cases of this scenario, the D-ZoSAP outperformed the baselines by far. Mainly, when we have highly skewed popularity (γ=1.2) and content size of μ=2 GB, it outperforms by 102% BB-ZoMKP and increases to 125% at μ=16 GB. This is achieved because the D-ZoSAP strategy places contents at MH where they are most 'demanded'. The BB-ZoMKP showed better performance than the Greedy-MKP since the algorithm can cache more contents and get an exact solution per multiple MHs, without being aware of spatial demand. The Greedy-MKP has the lowest performance since it can't deal with the size constraints of the problem.

Impact of popularity skewness on cache hit ratio

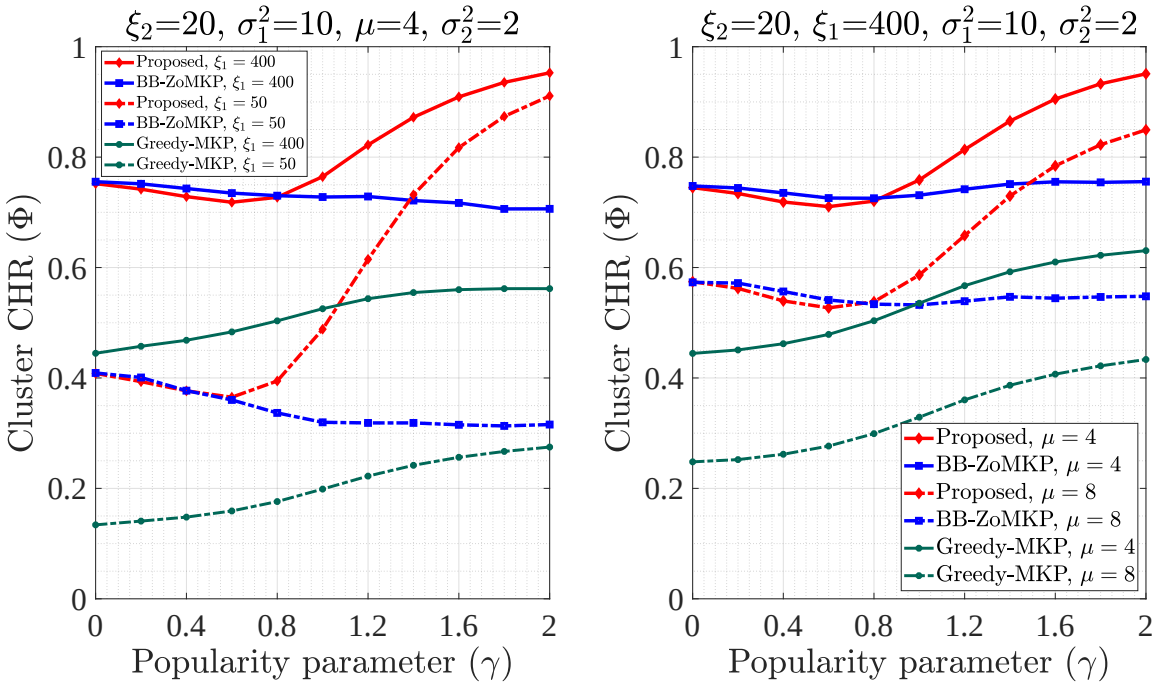

FIGURE 5.5: The CHR with increasing popularity index for a cluster of MHs: a) varying ξ_1 (left), b) varying μ (right).

The impact of the popularity index on cluster CHR is shown in plots of Fig. 5.5 for the global Φ of the cluster under: a) two cache size values of ξ_1=50 GB and ξ_1=400 GB, at μ=4 GB; b) two content sizes of μ=4 GB and 8 GB, at ξ_1=400 GB. The popularity index γ ranges from 0 (uniform popularity) to 2.0 (very few videos are very popular). In both cases, the FBS edges have cache mean size ξ_2=20 GB. The result shows that larger video contents have less CHR in the cluster for all strategies due to MHs cache

size limitation. This effect is noticed either when ξ_1 is reduced from 400 GB to 50 GB for fixed μ (left plot) or μ is increased from 4 GB to 8 GB for fixed ξ_1 (right plot).

Interestingly, the performance of the proposed strategy varies over a range of skewness levels. Looking at the left plot of Fig. 5.5, the D-ZoSAP performs almost equally with the BB-ZoMKP strategy at ξ_1=400 GB for $\gamma \leq$0.5. Even when we have sufficient cache size (*e.g.*, $\xi_1 \geq$400 GB) for $\gamma \leq$0.8, the BB-ZoMKP is preferred, however more complex and computationally costly. Based on mean cache size (μ), the D-ZoSAP gives an exponentially increasing CHR after some popularity index such as γ=0.5 for ξ_1=50 GB or γ=0.8 for ξ_1=400 GB. The CHR value for ξ_1=50 GB case drastically increases close to the ξ_1=400 GB case for γ=2. From this plot, we understand that γ has a significant relation with ξ_1, unlike the case of changing the content mean size, such as from μ=4 GB to μ=8 GB (right plot). In general, for real cellular networks, with $\gamma >$0.8, the proposed strategy is quite useful. This high performance is achieved because D-ZoSAP caches contents at serving MHs, through which they receive the highest demand rate from active users.

Though the BB-ZoMKP strategy gives high CHP performance [141] (only focuses on choosing very popular ones), its CHR slightly decreases across γ values because the γ seen in each MH is counterbalanced at the MBS, so the popularity is less affected. The strategy disregards the demand of individual MHs so that highly popular and impactful contents are displaced from their destined serving MH. In contrast, but with lower performance, the Greedy-MKP strategy increases with skewness because it places free video content at higher-demanding MH and succeeds in a cache hit.

Computation cost analysis

We analyzed the computation cost of the D-ZoSAP strategy by the time the algorithm takes to accomplish a single-round selection. The following two plots indicate the time: 1) under an increasing cluster cache size of ξ_2=0 GB (there is no SBS) up to 500 GB, ξ_2=20 GB, where γ=1.0, μ=4 and μ=8 GB; ii) under increasing content popularity indexes of γ=0.1 up to γ=2.0, where ξ_1=400 GB, ξ_1=400 GB, μ=4 GB, and μ=8 GB.

In Fig. 5.6, the computation time increases over the cache sizes except for all strategies except the Greedy-MKP. The cost of the proposed strategy is slightly higher than the Greedy-MKP because of extra iterations to assign contents at the most demanding MH. Still having great CHR performance, the proposed strategy's computation time is below 1 second for most cases and less costly than the BB-ZoMKP. From the plot, we notice that larger contents seem less costly to assign using the proposed strategy. However, the content size has an insignificant impact on the computation time.

Similarly, Fig. 5.7 indicates that the content popularity index almost does not affect the computation time for baseline strategies. Also, the impact of the content size is not that relevant. In contrast, the time for the proposed strategy goes up 2.6 seconds for γ=0.1 because the contents are nearly uniformly popular, which is not practically happening, so the number of iterations increases. For the same reason, smaller content sizes are computationally more demanding than bigger ones. Interestingly, the computation cost for the proposed strategy significantly drops over the increasing popularity index, *e.g.*, the computation time goes below 1 second for $\gamma \geq$1.0 and μ=8 GB. This

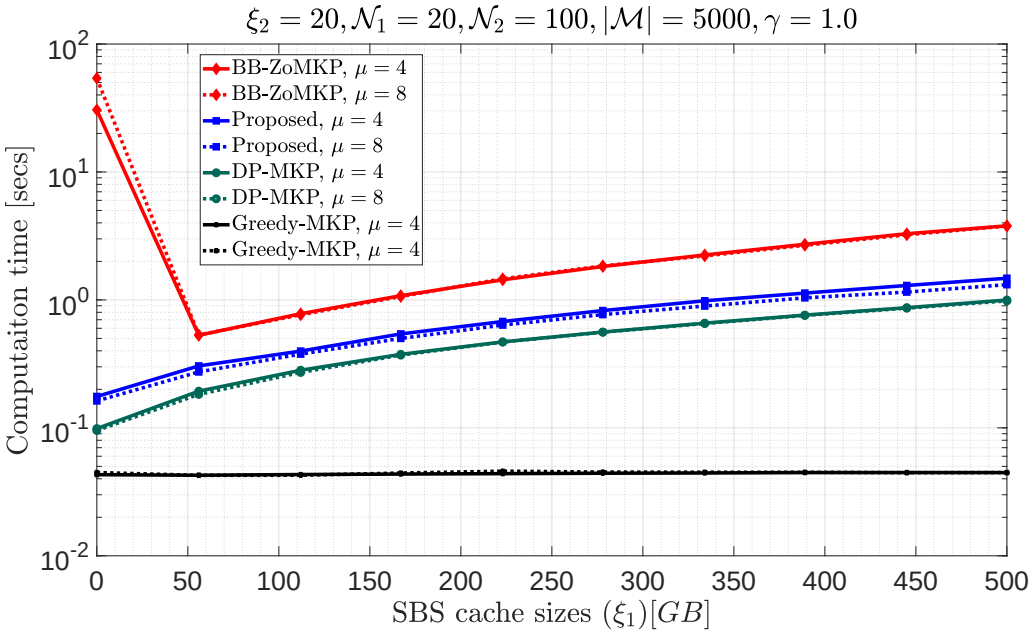

FIGURE 5.6: Computation time under increasing cluster cache size.

achievement is crucial to employ the strategy in real networks with highly skewed popularity indexes.

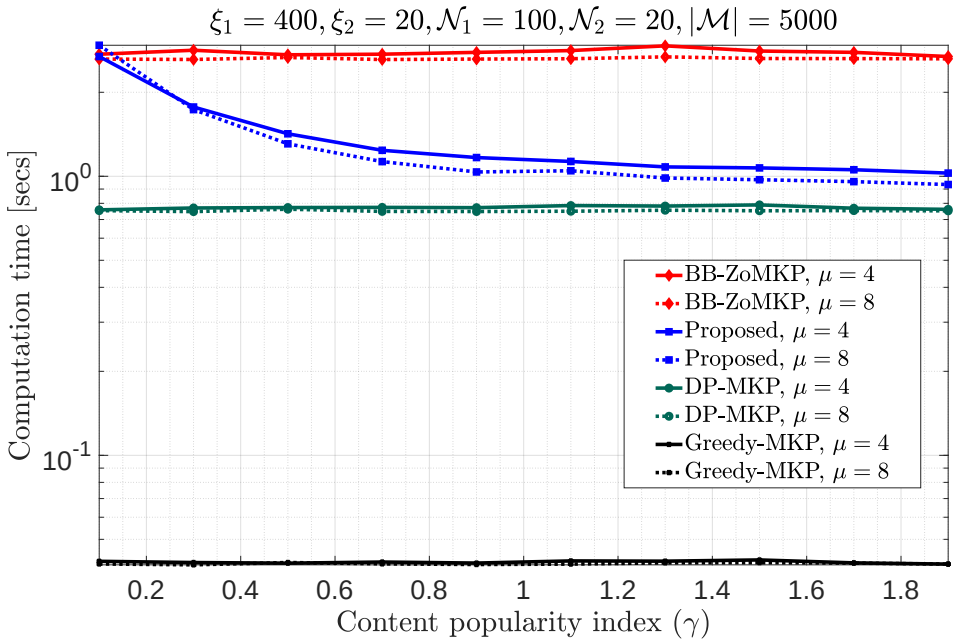

FIGURE 5.7: Computation time under increasing popularity index.

5.1.10 Section summary

In this section, we have studied a novel cooperative content caching system in a cluster of MEC-enabled edges, which are densely deployed to offload the central MBS. The most important aspect of this scenario is that contents have different popularity and request probability for each MH edge. We focused on assigning contents to the MHs where they are most 'demanded'. Using the combined impact of the local parameters of the request probability, the local popularity, and the spatial request ratio, we have modeled the caching scheme using the Separable Assignment Problem. We then proposed an iterative combinatorial content placement strategy, where an optimal solution is found for each MH, sequentially using DP, in each iteration. The outcomes of subproblems are merged to a filtering instance where overlapping contents are fixed to only the edge where they have maximum request probability.

The simulation results show that the proposed demand-based caching strategy outperforms baseline strategies. However, the performance is influenced by critical system parameters such as library size, cache size distribution, and content size and popularity distributions. Since the contents are found in its heavily requesting MH, the delivery cost is lessened. In addition, the MEC functionality in the enabled MHs makes RAN data analytics easier than the centrally-processed standard protocols so that the computation cost is highly minimized. This caching scheme can be scaled up to a joint caching to optimize the delivery cost at the cost of system complexity.

5.2 Cooperative Caching with Partitioning

This section details a caching strategy applied to a mutual concurrent scenario of three important models: multiple knapsack problems using the ZoMKP formulation, demand-aware caching using the SAP formulation, and content partitioning. Like in other scenarios, numerous MHs are clustered around a central MEC-enabled head, the MBS, which controls the caching process. The MHs try to maximize their benefit in this case, so they need a 'demand-aware' caching strategy.

5.2.1 Caching system model

The system model is similar to the one defined in Section 4.3, where the MBS has a list of video contents, defined as: $\mathcal{M}=\{f_m : 1 \leq m \leq |\mathcal{M}|\}$. All cache-capable SBSs and FBSs have different cache sizes and serve as MH. For each computation epoch, content popularity vector $\mathcal{P}=\{\rho_1, \rho_2, ..., \rho_{|\mathcal{M}|}\}$ is known (which holds $\sum_{m=1}^{|\mathcal{M}|} \rho_m=1$), that can be derived by any popularity estimation technique using the aggregate of request rate towards each content. Unlike the MKP, this problem tries to maximize the gain of each MH, so it becomes quite complex to reach any feasible solution. Again, unlike the MKP formulation, content partitioning is encouraged in this design. Hence, contents are cached by taking them entirely or fragmenting them into subcontents, also called *chunks (w)*, using any convenient compression algorithm.

With the permission of content partitioning, based on the request rates from the MHs, we devise a technique that provides a suboptimal result. For that matter, this formulation aims to select a set of chunks ($\mathcal{W}_n$), $\mathcal{W}_n=\{w_{m,n} : m \in \mathcal{M}, 1 \leq n \leq |\mathcal{N}|\}$, and used to populate each MH. However, the sum of their size should not exceed the cache size of the respective MH. Given library $\mathcal{M}$, $\mathcal{P}$, and $\mathcal{S}$, the CHP maximization problem is formulated in Eq. 4.2, subjected to constraints of Eq. 4.2b-4.2e. Because of its variety in partitioning, this problem is modeled by the Special Multiple Knapsack Problem (SMKP). It is logically represented in Fig. 5.8, with chunk placement format.

Once the caching process is completed and when a new request comes from a UE, the MHs strongly cooperate to deliver sufficient chunks to UEs through the appropriate path. Hence, the requesting UEs can decode the content from the chunks. If no trace of the content is found in any of the MHs, it is served from the MBS.

5.2.2 Proposed simplified method

This simplified content caching technique is a demand-aware MKP (D-MKP) strategy that has two independent steps: i) content selection by relaxation, to choose the most popular contents from $\mathcal{M}$ and, ii) content assignment, which assigns selected contents to the MHs based on their interest.

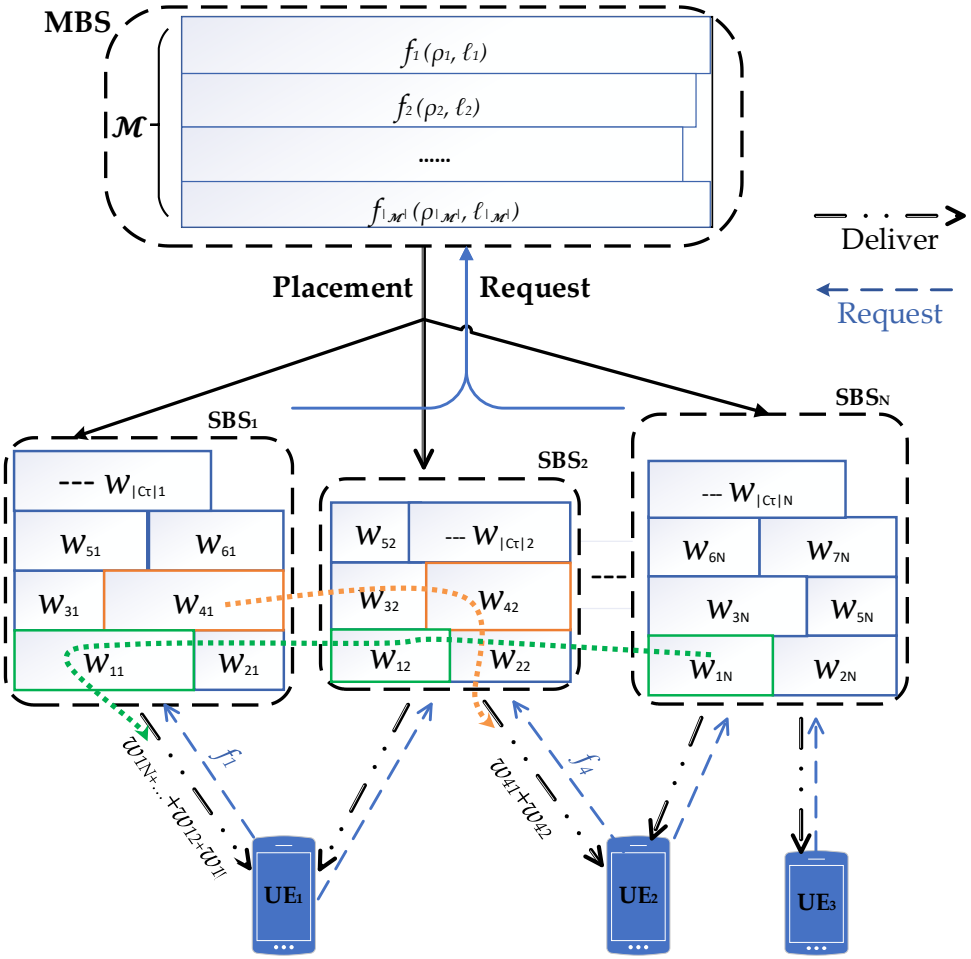

FIGURE 5.8: The MBS places non-overlapping chunks to cooperating MHs.

Content selection by relaxation

The straightforward task to maximize the problem $P_{4.1}$ (Eq. 4.2) is selecting the most popular contents from $\mathcal{M}$, assuming a single cache. Hence, we apply a combinatorial selection method at the MBS by first relaxing the constraint (4.2b) and getting the tightest upper bound. For that purpose, the surrogate relaxation, which is very practical for the case, is chosen in that: i) contents can be easily partitioned, and ii) there are many items to be placed at a few MHs. To apply the standard surrogate relaxation, let $(\pi_1, ..., \pi_n)$ be a vector of positive multipliers which satisfies: $\sum_{n=1}^{|\mathcal{N}|} \pi_n \sum_{m=1}^{|\mathcal{M}|} x_{m,n} \cdot s_m \leq$

$\sum_{n=1}^{|\mathcal{N}|} \pi_n \cdot L_n$, so that the ZoMKP formulation in Eq. 4.2 is redefined as:

$$P_{5.2}: \max_{x_{m,n} \in \mathcal{X}} \sum_{n=1}^{|\mathcal{N}|} \sum_{m=1}^{|\mathcal{M}|} \rho_m \cdot x_{m,n} \tag{5.6a}$$

$$\text{subject to:} \quad \sum_{n=1}^{|\mathcal{N}|} \pi_n \sum_{m=1}^{|\mathcal{M}|} x_{m,n} \cdot s_m \leq \sum_{n=1}^{|\mathcal{N}|} \pi_n \cdot L_n \tag{5.6b}$$

$$4.2c - 4.2e \tag{5.6c}$$

This relaxation has to select a set of cluster contents ($\mathcal{C}$) for further process. This relaxation is done by minimizing the upper bound of problem $\mathbf{P}_{5.2}$, which means the tightest UB. Hence, reducing the constraint violation. For any feasible solution with decision $x_{m,\bar{n}}$ such that: $\bar{n} = \arg\min\{\pi_n : n \in \mathcal{N}\}$, constraint (5.6b) is equivalent to: $\sum_{m=1}^{|\mathcal{M}|} x_{m,\bar{n}} \cdot s_m \leq \lfloor \sum_{n=1}^{|\mathcal{N}|} \frac{\pi_n \cdot L_n}{\pi_{\bar{n}}} \rfloor$. This inequality is required to be minimized and since $\sum_{n=1}^{|\mathcal{N}|} L_n \leq \lfloor \sum_{n=1}^{|\mathcal{N}|} \frac{\pi_n \cdot L_n}{\pi_{\bar{n}}} \rfloor$, any positive constant π gives minimum capacity. As proved in [133], this is the optimal vector of multipliers that gives the tightest upper bound, *i.e.*, when $\pi_1 = \cdots = \pi_n = \pi > 0$. Hence, taking any constant π and setting $y_m = \sum_{m=1}^{|\mathcal{N}|} x_{m,n}$, problem (5.6) is further reduced to a ZoSKP as follows:

$$P_{5.3}: \max_{y_m \in \mathcal{Y}} \sum_{m=1}^{|\mathcal{M}|} y_m \cdot \rho_m \tag{5.7a}$$

$$\text{Subject to:} \quad \sum_{m=1}^{|\mathcal{M}|} y_m \cdot s_m \leq \sum_{n=1}^{|\mathcal{N}|} L_n \tag{5.7b}$$

$$y_m \in \{0,1\}, 1 \leq m \leq |\mathcal{M}| \tag{5.7c}$$

The constraint in Eq. 5.7b is the cluster cache size. The simplified placement problem $\mathbf{P}_{5.3}$ in Eq. 5.7 can be optimally solved using the DP-ZoSKP Algorithm (Section 3.4). We allow the partitioning of selected contents into fitting chunks so that it fills all MHs cache spaces. Hence, this objective function is almost equal to the optimal value of the ZoMKP formulation in Eq. 4.2.

Balanced content assignment to MHs

The MHs have strong cooperation to deliver chunks to UEs, and each content has unique global popularity ρ_m at the MBS. Once we optimally select the candidate contents (get the $\mathcal{C}$ from the above step), we can partition them using any coding scheme and assign chunks to the MH caches.

(a) *Minimized contents' partitioning*: we sequentially choose a set of contents to place into all MHs. After fitting as many contents as possible to a cache L_n, the overhead portion of the first breaking content (f_b), which does not fit the remaining space, *i.e.*, $f_b = min\left\{b : \sum_{m=1}^{b} s_m > L_n\right\}$. Hence, the residual space of L_n is filled

by partitioning f_b into two chunks. That means, a size portion of $s_b - (L_n - \sum_{m=1}^{b-1} s_m)$ and respective popularity portion of $(L_n - \sum_{m=1}^{b-1} s_m) * \rho_b / s_b$ is added to the L_n and the remaining portion is placed at L_{n+1}. This process continues to all MHs, but no content breaking for $L_{|\mathcal{N}|}$ because we have no space for the overhead chunk of the last content. Thus, the maximum number of contents that will be partitioned is $|\mathcal{N}|$-1. This number can be further reduced by *re-selecting* combinations whose sum of content sizes closely fits with L_n, using the Subset-sum Problem (SSP), as follows.

$$P_{5.4} : \max \sum_{m=1}^{|\mathcal{C}|} s_m x_m \tag{5.8a}$$

$$\text{subject to: } \sum_{m=1}^{|\mathcal{C}|} s_m x_m \leq L_n \tag{5.8b}$$

$$x_m \in \{0,1\}, f_m \in \mathcal{C}, m = 1,2,...,|\mathcal{C}|, n = 1,...,|\mathcal{N}| \tag{5.8c}$$

Since we take only exactly fitting subsets, we take the equality sign in the constraint (5.8b), and an optimal solution is obtained using DP. While using the SSP, if all contents are assigned to MHs, no content is partitioned. This combination gives the optimal solution to the main problem. However, the SSP minimizes cache space loss (reducing residual buffer), suppressing the individual interests of MHs towards each content, and the traffic load will be unbalanced.

(b) *All contents partitioning*: we partition almost all contents to $w_{m,n}$, as shown in Fig. 5.8. For that, we use the ratio of the number of all requests to every content. Further assumptions: i) partitioning content f_m in terms of its size has the same partitioning effect on its popularity, ii) there is no redundant request to a single f_m from the same UE through different MHs. Let $r_{m,n}$ be the number of requests to f_m through n^{th} MH in a specific epoch. Then, the total request to f_m is r_m=$\sum_{n=1}^{|\mathcal{N}|} r_{m,n}$ at the MBS. The contents are partitioned using a rate κ_m, κ_m=$r_{m,n}/r_m$ where $\sum_{n=1}^{|\mathcal{N}|} \kappa_m$=1, and determine chunk parameters: size ($w_{m,n}$), popularity ($\hat{\rho}_{m,n}$), and decision parameter ($\hat{y}_{m,n}$). A proportional popularity vector $\mathcal{P}_n = \{\hat{\rho}_{1,n}, \hat{\rho}_{2,n}, ..., \hat{\rho}_{m,n}\}$ is created for each n^{th} MH chunks $w_{m,n}$. Hence, the

demand-aware placement per MH is formulated as follows:

$$P_{5.5} : \max \sum_{m=1}^{|\mathcal{C}_n|} \hat{y}_{m,n} \cdot \hat{\rho}_{m,n} \tag{5.9a}$$

$$\textbf{Subject to:} \sum_{m=1}^{|\mathcal{C}_n|} \hat{y}_{m,n} \cdot w_{m,n} \leq L_n \tag{5.9b}$$

$$\sum_{n=1}^{|\mathcal{C}_n|} \hat{\rho}_{m,n} = \rho_{m,n} \tag{5.9c}$$

$$\sum_{n=1}^{|\mathcal{C}_n|} w_{m,n} = s_m \tag{5.9d}$$

$$\sum_{n=1}^{|\mathcal{C}_n|} \hat{y}_{m,n} = \{0, 1\} \tag{5.9e}$$

The D-MKP, with an 'all contents partitioning' scheme, allows UEs to get the dominant chunk of content from the MH, where it is most popular. On the contrary, another MH might cache no chunk because it did not receive any request. This chunk assignment indicates that the D-MKP strategy is demand-aware, which brings a fair availability of chunks at MHs. Upon request from users, all chunk-hosting MHs cooperatively serve them to the UEs through a convenient protocol.

5.2.3 Numerical results and discussion

The performance of the proposed placement strategy (*i.e.*, DP-MKP with all content partitioning) is evaluated by comparing it with iterative baseline strategies. The MHs are sorted in increasing order of their cache sizes to avoid bias in all strategies. For easiness, system parameters are: $|\mathcal{M}|$=100, γ=1.0, and s_m is exponentially distributed with μ=1.23 Gb (corresponds to a 4-minutes video [125] with 720p resolution, at a rate of 5 Mbps), $|\mathcal{N}|$=3. The cluster cache size (L) ranges from 10 Gb to 100 Gb. Wherein the MHs' cache sizes share is: L_1=0.2 $\cdot$ L, L_2=0.35 $\cdot$ L, and L_3=0.45 $\cdot$ L.

Performance in terms of CHP

The two plots in Fig. 5.9 evaluate the performance of the three algorithms over the relaxed problem $\mathbf{P}_{5.3}$ (*i.e.*, Eq. 5.7). Since we can partition contents and use the entire L, using this D-MKP strategy, a nearly optimal solution is found for the ZoMKP formulation, a reference for further analysis.

Proving the performance of DP (left plot), the right plot in Fig. 5.9 depicts that the proposed D-MKP caching strategy outperforms the other three baseline strategies. The baseline strategies sequentially fill the caches, without content partitioning, using methods: i) the DP algorithm (for DP-ascending), ii) greedy selection (for Greedy-ascending), and iii) random selection (for Random-ascending). It is worth noting that

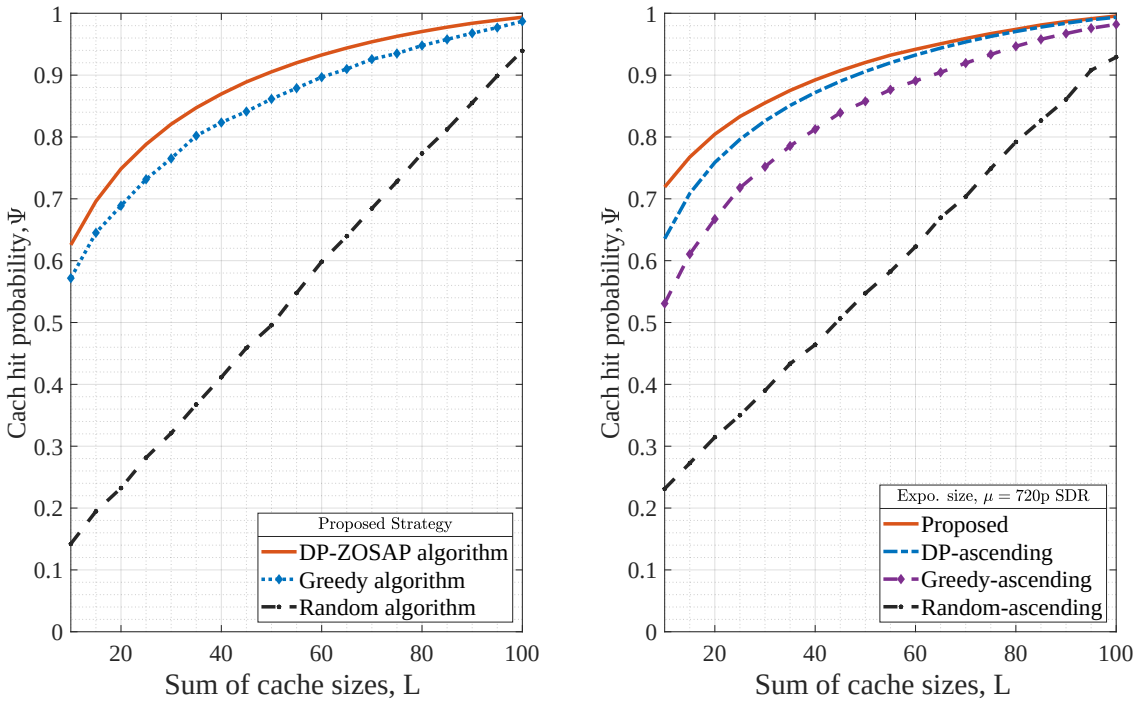

FIGURE 5.9: CHP performance: a) relaxed problem solved with three methods (left), b) D-MKP compared to baselines (right).

the D-MKP is even better than DP-ascending because it can break contents and place them in cache buffers. But when the cache sizes are relatively larger, also the baselines perform well because there is enough space for caching more contents.

Performance in terms of load balancing

The result in Fig. 5.10 compares the impacts of the partitioning in the D-MKP itself. The CHP values for each MHs are close to each other when we make 'all partitioning', unlike in the case of the 'minimized partitioning' scheme. This shows that the D-MKP with chunks synthesized based on the request ratio (κ) caches contents in a fair fashion. In contrast, the minimized partitioning scheme, which breaks less than $|\mathcal{N}|$ contents, incurs a very wide performance gap among the MHs. This shows that the contents are biased to some MHs, and the performance depends on the MH cache size and the order of filling. Interestingly, the performance gap between MHs for the proposed assignment method is consistent with increasing the $\mathcal{L}$ because the parameters of the assigned chunks are delimited by the independent request profile from each MH.

Performance in terms of service reward

The service reward is the amount of data downloaded, or the required data capacity, at the instant of serving a request. Since the popularity distribution is known and contents have no redundancy in the cluster, we can adapt the average service reward

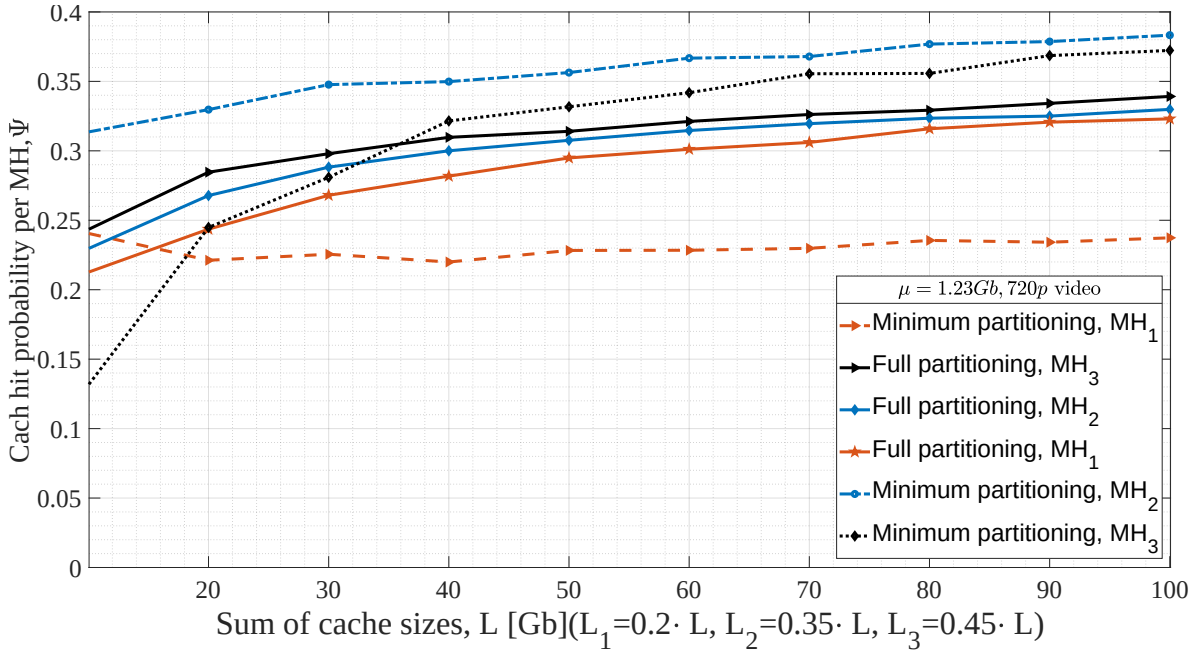

FIGURE 5.10: The D-MKP performance on CHP load balance, with different content partitioning schemes.

(Ω) of all cached contents from [30], as follows.

$$\Omega_{\mathcal{M}} = \sum_{n=1}^{|\mathcal{N}|} \sum_{m=1}^{|\mathcal{C}_n|} \hat{\rho}_{m,n} \cdot w_{m,n} \tag{5.10}$$

The plot in Figure 5.11, by computing Eq. 5.10, depicts that the D-MKP with a 'full partitioning' scheme highly reduces the link capacity budget by a factor of $1/3$ to serve the contents for each MH. In this case, the total Ω is very high (which means the downloading requirement is reduced) due to the ability to partition contents by a scale of not more than $|\mathcal{N}|$. In addition, the proposed strategy fairly balances the capacity load to the edges, in contrast to the 'minimized partitioning' scheme. Harnessing this straightforward strategy reduces the backhaul load of cellular networks, particularly when a content placement happens over a congested backhaul or during delivery.

5.2.4 Section summary

In this section, we have studied a content caching strategy where multiple network edges can cooperatively cache contents with freedom of partitioning. The content caching is modeled by special generalized knapsack problems that allow item partitioning, and it is reduced to a single knapsack problem using standard relaxation. The DP optimally solves the relaxed problem while the candidate contents are synthesized into chunks and assigned to MHs based on the request rates. This scheme increases the availability of chunks, or the entire content, at the MHs in a balanced

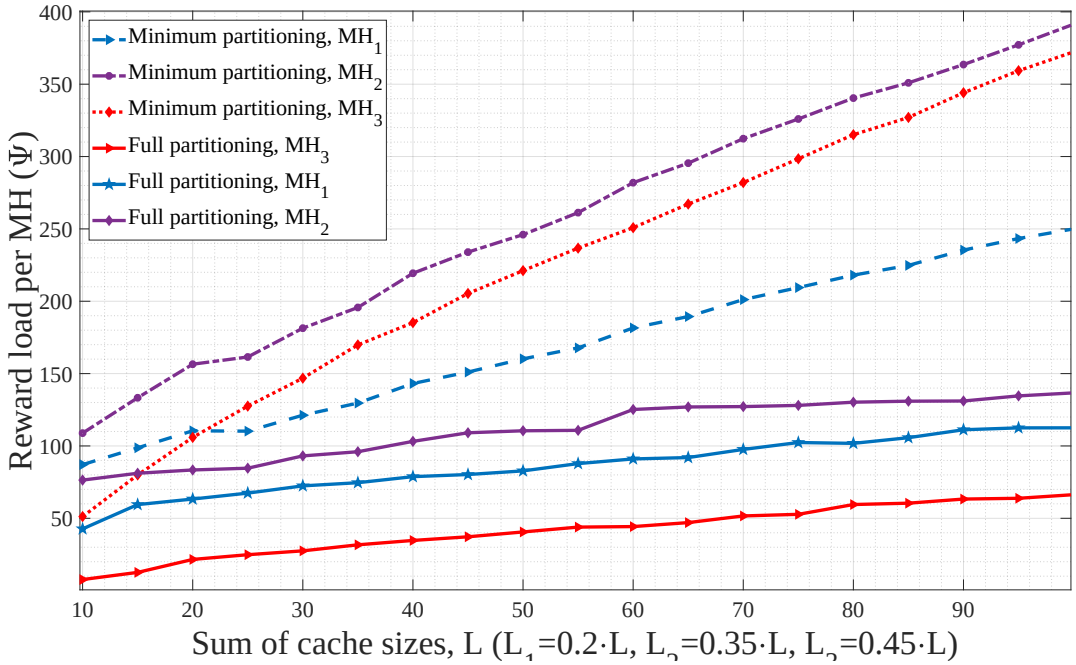

FIGURE 5.11: Serving reward for chunk assignment strategies per MH.

fashion of the service load. When necessary, the placement process can be updated with the computational capacity of caching edges. Such straightforward and very efficient optimization methods boost the performance of both the content caching process and the enabling technologies. The proposed D-MKP caching strategy needs multipath for synchronous transmission, so it is worth integrating the D-MKP strategy with multipath caching, prompted in Subsection 6.2.1.

5.3 Demand-aware Joint Content Caching

This section explains an outstanding extension to Section 5.1, where the caching process is to be optimized in an end-to-end approach. This section presents a joint caching formulation, mainly focusing on the delivery phase, with appropriate modifications to the content placement formulation. This section's relevant system model is the same as the design detailed in Subsection 5.1.2. The system's MEC computation processes and radio information analytics are explained in Subsections 5.1.3 and 5.1.4, respectively.

5.3.1 Introduction

A *joint caching* procedure is a method where the content placement and delivery phases are mutually optimized using an end-to-end approach. This scheme is extremely complex because the two phases have a strong relationship, each with a different objective. Their relationship is that the MBS (*i.e.,* the *leader*) makes its own content placement decision to maximize the content availability, in the cluster, by anticipating the possible gains of the content delivery decision. On the other hand, the MHs (*i.e.,* the *followers*) make their own delivery decision, after the leader's placement decision. Followers have the objective of minimizing service utility costs. Hereof, the objective of the entire MEC system is to maximize the CHR by the leader, subject to a layer of constraints, and minimize delivery cost by followers, subject to their constraints.

So far, several joint caching approaches anticipated end-to-end optimization. The authors in [12] decouple the original joint content placement and delivery problem: i) using integer linear programming for solving the content placement problem and ii) employing unbalanced assignments for solving the content delivery problem. However, they rely on a single-level decision by moderating the placement problem into a delivery problem in an intractable formulation. Another remarkable work in [10] formulates a joint content and delivery scheme as a nested dual optimization problem, where the two phases are strongly interdependent. Accordingly, authors relax the original *nested dual* problem to a mixed integer non-linear program (MINLP) and employ a branch-and-bound method to enable MHs to decide independently. Although the respective formulation and methodology are equipped with strong mathematical formulations, the proposed system-wide centralized optimization faces significant scalability challenges when having many MHs and centralized aggregation of RAN information. Regardless of this limitation, authors claim that 'where to cache contents is influenced by delivery decisions while how to deliver them is impacted by the content placement', which is a slip to HCN real property.

Acknowledging the mathematically sounding relationship of the two caching phases, we argue that the content delivery phase does not strongly influence the content placement due to the following characteristics of caching: i) the content placement is usually done at an off-peak time, and the decision is expected to wait for several minutes or hours (multiple epochs) while content delivery is an instant decision [122], ii) because the UEs are extremely dynamic, the placement decision is made using MEC analytics on past data and almost without knowing current statistics of the UEs, iii)

the content placement decision is information-centric while the delivery decision is a user-centric process, *i.e.*, making content placement decision based on its RAN data analytics is more effective than based on individual UE data. Therefore, treating the content caching problem as a dual-nested problem is ineffective modeling. Rather, using hierarchical decisions by the leader and the follower players is much better.

In our novel approach, we assume a weak influence of the content delivery decision over the content placement decision. This is only because the leader has to foreknow the outcome of followers' actions. That means the delivery problem is an embedded constraint to the content placement problem. Once the contents are optimally placed at the cooperative network edges, each serving MH decides on its delivery path to users at a minimized service cost for each lively requested content.

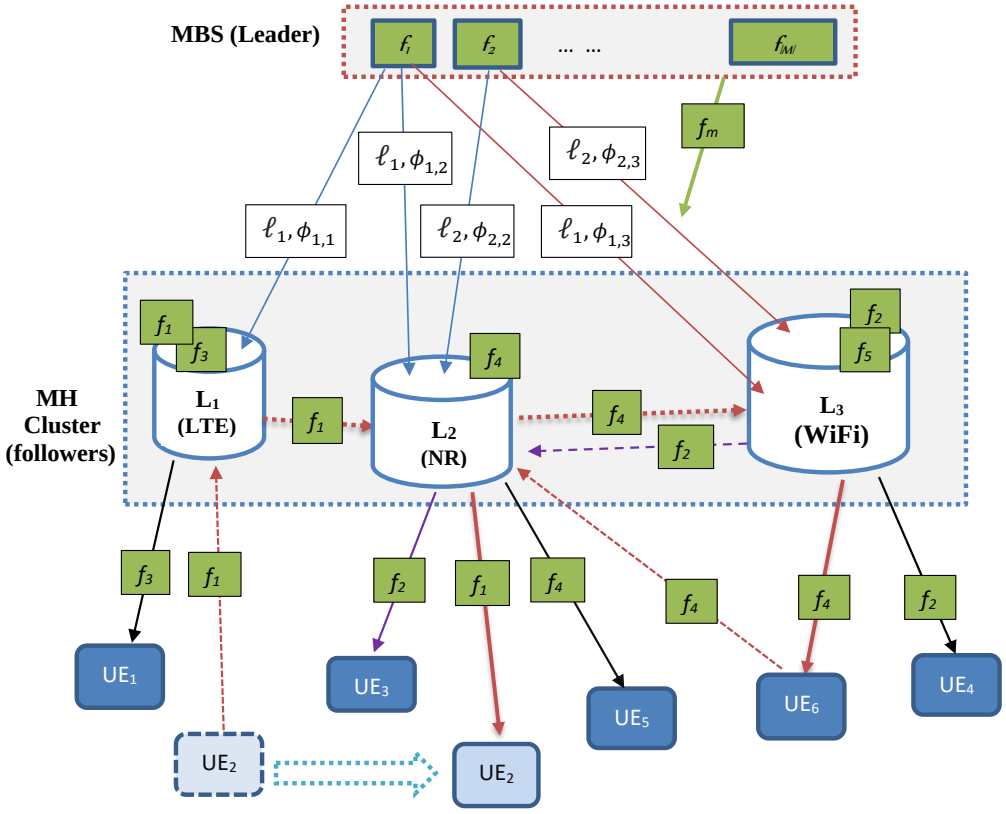

FIGURE 5.12: Joint content placement and delivery scheme.

5.3.2 Content delivery scheme

Due to the heterogeneous nature of the 5G architecture, the computational complexity associated with caching is not trivial. Far from its simple representation in Fig. 5.12, contents are distributively placed in different network types: some are in the 5G-NR, some are in the 4G-LTE, and some other contents might be in the WiFi network [144]. On top of that, due to the very dynamic nature of users, they may change their profile after requesting content. For example, they move away (like UE_2 moved from MH_1 to

MH_2), switch to another network (like the UE_2 switched from the 4G LTE network to the 5G NR), or change their path. When necessary and cognizant of applied handover policies, the MEC functionality determines a user-to-MH association by applying utility minimization approaches. Thus, the requesting UE might be re-associated to a new local MH (recall, also called *local MH*), and the central MBS is informed. In such a case, one UE may be associated with different MHs for the uplink and downlink services [80]. That allows the MEC system to establish a new downlink path and directly serve contents from caching MH to the requesting UE.

In the case of content delivery, the intelligent MHs in the MEC system determine the transport routes that reduce the cluster cost. Again having a close look into Fig. 5.12 and referring to Fig. 5.2, and the $x_{m,n}$ as an indicator of content availability, the *content delivery* phase is explained as follows. Any UE requests content (f_m) through its associated local MH and, by checking from the lookup table (*i.e.*, the MH compute process ① in Fig. 5.2), one of the following three events occurs: i) a *cache hit event* happens, means that f_m is found at local MH ($x_{m,n}$=1), with access probability $\phi_{m,n}$ thus f_m is immediately delivered to u, ii) a *local cache miss* happens, means that f_m is not found in its cache ($x_{m,n}$=0) but at a neighboring helper k ($x_{m,k}$=1, $k \neq n$), with access probability $\phi_{m,k}$. In this case, the local MH forwards the request ID and location information ($info$) to hosting edge k, so the content is served through the best MH [122], which delivers f_m to u through the established downlink, iii) a *cache miss* happens ($x_{m,n}$=0, $\forall n \in \mathcal{N}$), which means that f_m is not found in the cluster, with access probability (1-$\phi_{m,n}$-$\phi_{m,k}$). For this case, the local MH forwards the request and $info$ to the central MBS. So, f_m is directly downloaded from the MBS to the associated MH. In this communication protocol, since the 5G NR architecture uses the Control and User Plane Separation (CUPS) strategy [145], the upper-level packet processing and data aggregation are evenly distributed to network edges and ease packet forwarding.

Delivery utility cost analytics

Utility cost ($v_{m,u}$) refers to the content serving cost incurred when delivering f_m in the downlink after being requested through local MH. The cost may include time taken and power consumption, which depend on various physical-layer factors such as edge density and the serving distance between u and MH.

For a requesting user (u), the cache decision variable $x_{m,n}$ indicates the availability of content f_m at its local MH (n). Let $x_{m,k}$ refer to content availability at any neighboring MH. Thus, corresponding with the above three cache events (subsection 5.3.2), the delivery cost model is given as follows:

$$v_{m,u} = \begin{cases} v^n_{m,u}; & \text{if} \quad x_{m,n} = 1 \\ v^k_{m,u}; & \text{if} \quad \bigcup_{\substack{k=1 \\ k \neq n}}^{|\mathcal{N}|} x_{m,k} = 1 \\ v^M_{m,u}; & \text{if} \quad \bigcup_{n=1}^{|\mathcal{N}|} x_{m,n} = 0 \end{cases} \tag{5.11}$$

where $\bigcup$ is operator such that $\bigcup_{n=1}^{|\mathcal{N}|} x_{m,n} = x_{m,1} \oplus x_{m,2} \oplus ... \oplus x_{m,|\mathcal{N}|}$. In Eq. 5.11, the notation of instantly delivering cost ($v_{m,u}$) varies based on the edge type. Here, cost

segment $v^n_{m,u}$ occurs when content is delivered from the local MH (*i.e.*, $f_m \in \mathcal{C}_n$) to the user while $v^k_{m,u}$ is incurred when it is served from hosting MH (*i.e.*, $f_m \in \mathcal{C} \setminus \mathcal{C}_n$), including forwarding cost ($v^f_{n,k}$). This $v^f_{n,k}$ is the cost of MEC signaling overhead to k and forwarding cost to the serving MH. Lastly, the cost segment $v^M_{m,u}$ is incurred when the content is served from the MBS, *i.e.*, $f_m \in \mathcal{M} \setminus \mathcal{C}$.

The total content delivery cost of the system is estimated by taking the sum of all service costs for each u through all three events and associated costs. In this regard, we consider contents requested only through respective local n, whose event status is known by $x_{m,n}$ and $\{x_{m,n}\} \subseteq \mathbf{x}$. Let $y_{m,u}$ indicate that f_m is served to u by its serving edge, regardless of request route, such as $y^n_{m,u}$ is for local MH, $y^k_{m,u}$ is for other hosting MH, and $y^M_{m,u}$ is for the MBS, where $y^n_{m,u} \in \{0,1\}$. Let also $\tilde{\mathcal{U}}_n$ represent the set of UEs, among active UEs associated with the MH, that instantly request f_m. It is worth noting that a single f_m might be simultaneously served to many UEs from the hosting MH. Given the cache hit event access probability $\phi_{m,n}$ and cost $v_{m,u}$; defined on libraries $\mathcal{M}$, $\mathcal{N}$, and $\mathcal{U}_n$, the total delivery cost of the system ($\Upsilon(\mathbf{x}, y^n_{m,u})$) is expressed in Eq. 5.12.

$$\Upsilon(\mathbf{x}, y^n_{m,u}) = \underbrace{\sum_{n=1}^{|\mathcal{N}|}\sum_{m=1}^{|\mathcal{M}|} x_{m,n}\cdot\phi_{m,n}\left[\sum_{u=1}^{|\tilde{\mathcal{U}}_n|} y^n_{m,u}\cdot v^n_{m,u}\right]}_{\text{First Case}} + \underbrace{\sum_{n=1}^{|\mathcal{N}|}\sum_{m=1}^{|\mathcal{M}|}\left(\bigcup_{\substack{k=1\\k\neq n}}^{|\mathcal{N}|} x_{m,k}\right)\cdot\phi_{m,k}\cdot\left[\sum_{u=1}^{|\tilde{\mathcal{U}}_n|} y^k_{m,u}\cdot v^k_{m,u}\right]}_{\text{Second Case}} + \underbrace{\sum_{n=1}^{|\mathcal{N}|}\sum_{m=1}^{|\mathcal{M}|}\left(1 - x_{m,n} - (\bigcup_{\substack{k=1\\k\neq n}}^{|\mathcal{N}|} x_{m,k})\right)\cdot\phi_{m,n}\left[\sum_{u=1}^{|\tilde{\mathcal{U}}_n|} y^M_{m,u}\cdot v^M_{m,u}\right]}_{\text{Third Case}} \tag{5.12}$$

$$= \sum_{n=1}^{|\mathcal{N}|}\sum_{m=1}^{|\mathcal{M}|}\sum_{u=1}^{|\tilde{\mathcal{U}}_n|} y^M_{m,u}\cdot\phi_{m,n}\cdot v^M_{m,u} - \sum_{n=1}^{|\mathcal{N}|}\sum_{m=1}^{|\mathcal{M}|} x_{m,n}\cdot\phi_{m,n}\left[\sum_{u=1}^{|\tilde{\mathcal{U}}_n|} y^n_{m,u}\cdot v^M_{m,u}\right] + \sum_{n=1}^{|\mathcal{N}|}\sum_{m=1}^{|\mathcal{M}|} x_{m,n}\cdot\phi_{m,n}\left[\sum_{u=1}^{|\tilde{\mathcal{U}}_n|} y^n_{m,u}\cdot v^n_{m,u}\right] - \sum_{k=1}^{|\mathcal{N}|}\sum_{m=1}^{|\mathcal{M}|}\left(\bigcup_{\substack{k=1\\k\neq n}}^{|\mathcal{N}|} x_{m,k}\right)\cdot\phi_{m,k}\left[\sum_{u=1}^{|\tilde{\mathcal{U}}_n|} y^n_{m,u}\cdot v^M_{m,u}\right] + \sum_{k=1}^{|\mathcal{N}|}\sum_{m=1}^{|\mathcal{M}|}\left(\bigcup_{\substack{k=1\\k\neq n}}^{|\mathcal{N}|} x_{m,k}\right)\cdot\phi_{m,k}\left[\sum_{u=1}^{|\tilde{\mathcal{U}}_n|} y^n_{m,u}\cdot v^k_{m,u}\right] \tag{5.13}$$

Here, the expression $\bigcup_{\substack{k=1\\k\neq n}}^{|\mathcal{N}|} x_{k,m} = x_{1,m} \oplus ... \oplus x_{|\mathcal{N}|,m}$ indicates the placement of content at unique MH, in the cluster MHs, except the local MH. Also note the value $\phi_{m,n}$ changes over the edge type. By the assumption of non-repetition of contents in the

cluster, $\bigcup_{\substack{k=1 \\ k\neq n}}^{|\mathcal{N}|} x_{k,m}$={0,1}. By the expansion of system service cost in Eq. 5.12, and sorting as shown in Eq. 5.13, we get the expression for the delivery utility cost in Eq. 5.14, which comprises two gains of $G_1(x_{m,n}, y^n_{m,u})$ and $G_2(x_{m,k}, y^k_{m,u})$.

$$\Upsilon(\mathbf{x}, y^n_{m,u}) = \underbrace{\sum_{n=1}^{|\mathcal{N}|}\sum_{m=1}^{|\mathcal{M}|}\sum_{u=1}^{|\tilde{\mathcal{U}_n}|} y^M_{m,u}\cdot\phi_{m,n}\cdot v^M_{m,u}}_{\text{Cost if all served from MBS}} \underbrace{- \sum_{n=1}^{|\mathcal{N}|}\sum_{m=1}^{|\mathcal{M}|} x_{m,n}\cdot\phi_{n,m}\left[\sum_{u=1}^{|\tilde{\mathcal{U}_n}|} y^n_{m,u}\cdot\left(v^M_{m,u} - v^n_{m,u}\right)\right]}_{G_1(x_{m,n}, y^n_{m,u})\text{: Gain from demand-aware caching}}$$
$$\underbrace{- \sum_{k=1}^{|\mathcal{N}|}\sum_{m=1}^{|\mathcal{M}|}\left(\bigcup_{\substack{k=1 \\ k\neq n}}^{|\mathcal{N}|} x_{m,k}\right)\cdot\phi_{m,k}\left[\sum_{u=1}^{|\tilde{\mathcal{U}_n}|} y^k_{m,u}\cdot\left(v^M_{m,u} - v^k_{m,u}\right)\right]}_{G_2(x_{m,k}, y^k_{m,u})\text{: Gain from cooperative caching}} \tag{5.14}$$

In Eq. 5.14, the first term indicates the delivery cost if all the contents are served from the central head, meaning that content caching is not applied. The second term $G_1(x_{m,n}, y^n_{m,u})$ indicates the utility gain from the demand-aware content caching scheme, served through all local MHs. Mainly, this gain is from the first sub-objective vector of $\left[v^M_{m,u} - v^n_{m,u}\right]$ values. The third term, $G_2(x_{m,k}, y^k_{m,u})$, indicates the gain when contents are accessed from the neighboring MHs in the cluster but not in their local MH. Therefore, this gain is entirely from the cooperative caching scheme, mainly attained from the sub-objective vector of $\left[v^M_{m,u} - v^k_{m,u}\right]$.

In the above formulation, Eq. 5.14, the required cost optimization of $\Upsilon(\mathbf{x}, y^n_{m,u})$ can be achieved by maximizing the two gains. However, it is not trivial because the two gains are also influenced by the accessibility of contents, which means the $\phi_{m,n}$. Therefore, maximizing the two gains becomes a binary nonlinear integer programming problem that can be treated with some solvers.

5.3.3 Joint caching formulation

The joint placement and delivery process is a nested and complex problem. To complement the duality nature of the joint caching process, we sorted into a bilevel optimization problem. To that end, we pivoted on optimizing the two objective functions by two decisions: maximizing the availability of contents by leader entity and minimizing its total utility cost $\Upsilon(\mathbf{x}, y^n_{m,u})$ by the follower entities.

Content placement decision by the leader

The MBS centrally makes decisions on content placement to maximize the availability of popular contents at the caching edges without violating their size limit. For that matter, we aim at selecting $|\mathcal{N}|$ number of non-overlapping subsets of contents $\mathcal{C}_n$, *i.e.*, $\mathcal{C}_n \cap \mathcal{C}_k = \emptyset, n \neq k, k \in \mathcal{N}$. These subsets are placed at appropriate MH to optimize the objective function globally. The upper-level content placement problem is

expressed in Eq. 5.5, where the non-repetition constraint in Eq. 5.5c is modified as: $x_{m,n} \oplus \bigcup_{\substack{k=1 \\ k \neq n}}^{|\mathcal{N}|} x_{m,k} \leq 1, \forall m : f_m \in \mathcal{M}$.

Content delivery decision by followers

Once contents are optimally placed at the caching edges, the system chooses the cost-effective path to deliver them to requesting UEs at a peak period. That means the central head must consider the delivery cost while making the content placement decision. This consideration shows that the optimization problem $P_{5.6}$, in Eq. 5.5, has an embedded objective function of globally minimizing the delivery cost. In addition, it is worth noting the assumptions that, at a given instant of delivery: i) several UEs might request for one content, ii) each UE gets only one content, iii) for any route choice and the number of transporting hops from caching MH to the UE, there is tolerable cost threshold ($\tilde{v}$), which we assumed to be equal for all subcarriers. Then, the cost minimization problem is given as shown in Eq. 5.15.

In Eq. 5.15, the first constraint (5.15b) shows that the number of active UEs served by each MH is limited by its subcarrier capacity threshold ($\tilde{y}_n$), pursuant to resource optimization in the RAN. The constraint (5.15c) indicates that the total utility cost, which is the consequence of the delay and energy consumption, for delivering content should be less than the acceptable cost threshold. Lastly, constraint (5.15d) shows that any UE is assigned and served by only one MH at any delivery moment [122].

$$P_{5.6}: \min_{\substack{\{x_{m,n} \oplus x_{m,k}\} \subseteq \mathbf{x} \\ f_m \in \mathcal{M}, \forall n \in \mathcal{N}, \forall u \in \mathcal{U}}} \Upsilon(x_{m,n}, y^n_{m,u}) \tag{5.15a}$$

$$\text{subject to:} \sum_{u=1}^{|\mathcal{U}_n|} y^n_{m,u} \leq \tilde{y}_n, \forall n \in \mathcal{N} \tag{5.15b}$$

$$\sum_{u=1}^{|\mathcal{U}_n|} v_{m,u} \leq \tilde{v}, \forall n \in \mathcal{N} \tag{5.15c}$$

$$\sum_{n=1}^{|\mathcal{N}|} y_{m,n} \leq 1, \forall m : f_m \in \mathcal{M} \tag{5.15d}$$

Recall that the main objective function $\mathbf{P}_{5.1}$ typically represents the placement optimization while the nested function $\mathbf{P}_{5.6}$ refers to the delivery optimization. Thus, the joint caching expression is given as follows, where placement and delivery phases are

mutually optimized:

$$P_{5.7}: \max_{\forall x_{m,n} \in \mathbf{x}} \Phi(\mathbf{x}, y^n_{m,u}) \tag{5.16a}$$

$$\text{subject to: 5.5b-5.5f} \tag{5.16b}$$

$$\min_{\substack{\{x_{m,n} \oplus x_{m,k}\} \subseteq \mathbf{x} \\ f_m \in \mathcal{M}, \forall n \in \mathcal{N}, \forall u \in \mathcal{U}}} \Upsilon(\mathbf{x}, y^n_{m,u}) \tag{5.16c}$$

$$\text{subject to: 5.15b-5.15c} \tag{5.16d}$$

In Eq. 5.16, the set of constraints in (5.16b) are system limits for the cache placement process. In contrast, constraint (5.16c) is incorporated to anticipate the optimal decision by the follower entity. Generally, any placement decision without considering the nested delivery optimization will bring higher utility costs and deteriorated QoE. Again, the constraints in (5.16d) act as second-order limitations, but the placement is not directly limited. Since the placement decision tries to assist the delivery decision, they have an optimistically nested relationship.

The problem formulated in Eq. 5.16 is mapped to the bilevel optimization problems (BOP), and thus, let us call the caching method as *demand-aware joint* caching using the BOP strategy (DJ-BOP). Although the BOP outstandingly represents joint caching, this problem is very complex to solve. However, there are several attempts in economics and defense areas of operational research that solve them near-optimally. The majority of the techniques lie in the game theory, and reformulations by Extended Mathematical Programming (EMP) approaches. Specifically, this BOP is reformulated to mathematical programs with equilibrium constraints (MPEC) and can be solved using one of the non-linear solvers, such as general algebraic modeling systems (GAMS).

5.3.4 Numerical results and discussions

In this subsection, we evaluate the performance of the proposed DJ-BOP strategy in terms of energy consumption for making content delivery. In this analysis, we limit the simulation of the joint caching: i) applying the D-ZoSAP strategy (Section 5.1.8) for the leader decision process, and ii) using the cost estimation in Eq. 5.14 for followers' delivering process. Its energy cost is compared to the cost incurred while using baseline strategies, which are not demand-aware. The cost of baseline strategies is due to: i) *BB-ZoMKP*, a caching strategy that applies the MKP for modeling and DP for solving each iteration; ii) *Greedy-MKP*, a caching strategy that applies MKP for modeling and selects contents in decreasing order of their popularity. Unlike the DJ-BOP strategy, the global popularity of each content is similar to all MHs in the case of baseline strategies.

The simulation setup is basically similar to the one used in subsection 5.1.9 while numerical parameters described in Table 5.1 are assumed [146], [147]. However, these critical parameters need experimental justifications. The additional assumption here is that the $\mathcal{U}_n$ follows a normal distribution with mean UE density of δ while each content receives requests from all MHs based on Poisson distribution, with mean request λ. For better understanding, the simulation is made for all files in $\mathcal{M}$ where all instantly

active UEs request to a content. Both cases lead to a high total system cost. The essence is that we want to analyze the delivery cost if all of the contents are requested. Note that a single content might be served to several UEs.

According to the utility model in Eq. 5.11, the unit cost of serving contents to requesting UEs is estimated as the energy consumption [Joule/bit] based on how they are delivered. If the content is delivered: i) from local MH, the consumption will be only e_n; ii) from other hosting MH, the cost will be $(e_k + e_n + e_{fk} + e_{lk})$, iii) from central MBS, the unit consumption will be $(e_c + e_{fm} + e_{lc} + e_n)$. For the two cases, the request forwarding and link costs are added. In the last case, *i.e.*, when a cache miss event happens, contents are served from the MBS through at least one serving node, which instantly becomes local MH. Note that the utility cost directly depends on the size of contents and frequency of requests in a given epoch, so the unit costs are multiplied with s_m and $r_{m,n}$.

TABLE 5.1: MEC task processing simulation parameters

Parameter description	**Values**
Content processing energy density by local MH (e_n)	2.0×10^{-8} J/bit
Content forward energy density by neighboring MH (e_k)	2.0×10^{-9} J/bit
Content processing energy density by a central MBS (e_c)	4.0×10^{-8} J/bit
Content forwarding energy density to hosting MH (e_{fk})	1.5×10^{-10} J/bit
Content forwarding energy density to central MBS (e_{fc})	4.5×10^{-9} J/bit
Content serving energy density of a link from central MBS (e_{lc})	4.5×10^{-7} J/bit
Content serving energy density of a link from hosting MH (e_{lk})	1.5×10^{-9} J/bit
Mean number of actively requesting UEs (δ)	10, 20
Mean request rate to a content (λ)	100

As explained in the following two subsections, the analysis focuses on the impact of certain parameters such as caching size, content size, content popularity index, and user density on system cost reduction. It also shows the fundamental utility cost gain of content caching in the RAN architecture.

Impact of content and cache sizes on delivery cost

This analysis increases the cluster cache size by spanning the ξ_1 from 0 to 500 GB, ξ_2 is set to 20 GB, under two mean content size values. As can be seen in Fig. 5.13, the utility cost decreases due to caching technology because the contents are placed very close to the user. More importantly, increasing cluster cache size gives higher cost relief to the system. Again, contrasting the two sets of plots shows that smaller contents, μ=4 GB, cost much lower than larger contents, μ=8 GB. In general, caching smaller contents in a large caching system gives multiple-fold delivery cost reduction.

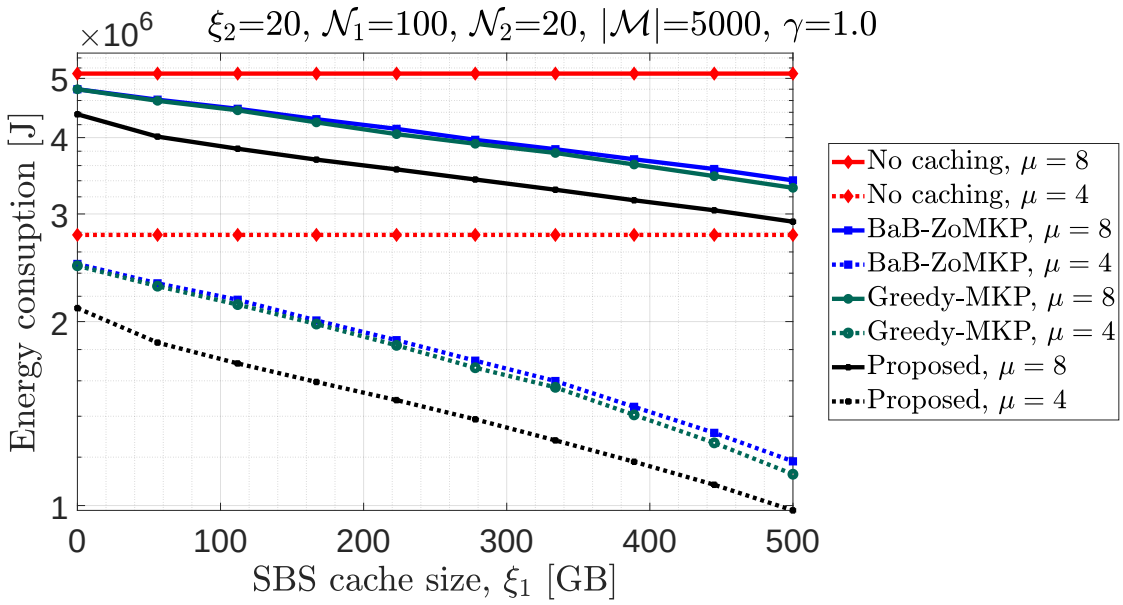

FIGURE 5.13: System utility cost for an increasing cluster cache size.

In the same plot, we observe that the proposed strategy is more effective in cost reduction than baseline strategies for all scenarios. This is because the D-ZoSAP algorithm can place contents in the most requesting node, while other strategies cache contents without prioritizing content demand. In contrast, the two baseline strategies are almost equally costly unless we have smaller size contents, whereas the Greedy-MKP slightly outperforms because it takes short but unpopular files.

Impact of UE density and popularity index on delivery cost

In this analysis, the content popularity parameter (γ) is spanned from 0 (uniform popularity) to 2 (very highly skewed popularity) under two δ values, where γ values are similarly applied for each MH. Other system parameters are fixed to identify the impact of the content popularity index and the active user density. Hence, the active UEs are exponentially associated with each serving MH: $|\mathcal{U}_n| \simeq exp(\delta)$, where δ=10, 20.

From the plots in Fig. 5.14, we observe that content caching relieves the system cost by more than half. This is more important when the popularity index increases, which leads to more frequent fetching of popular contents from the central system. In addition, having highly dense active UEs is more costly, except for higher γ in the case of DJ-BOP. For all cases of baseline strategies (and $\gamma \leq 0.7$ for DJ-BOP), doubling the density of active UEs, say from δ=10 to δ=20, leads to doubling the delivery cost. This is because active users increase content requests' frequency and redundancy, reducing repeated content serving to the active UEs.

All caching strategies equally perform in delivering cost optimization when contents are nearly uniform in popularity, such as $\gamma \leq 0.6$. The utility cost of baseline

strategies is almost unaffected by γ because they place contents to MHs based on aggregate popularity. But the cost of the DJ-BOP strategy deteriorates when the popularity index increases, which is the real network property. The proposed strategy is extraordinarily important when we have highly popular files. This achievement is because demand-aware caching can place contents at the MHs, where many active users can access them easily. Interestingly, the impact of user density diminishes when contents have skewed popularity distribution because the cache hit ratio increases.

We also observe that the Greedy-MKP strategy very slightly outperforms the BaB-ZoMKP strategy. This outcome is because the greedy approach only optimizes caching process based on content popularity, while BaB-ZoMKP optimizes based on popularity and size. It means that, though the BaB-ZoMKP strategy performs better in CHR, it places many contents in neighborhood MHs, creating a higher intra-MH link cost. This intuitive observation recommends applying the BaB-ZoMKP strategy for the joint caching optimization by taking content delivery cost as the objective function.

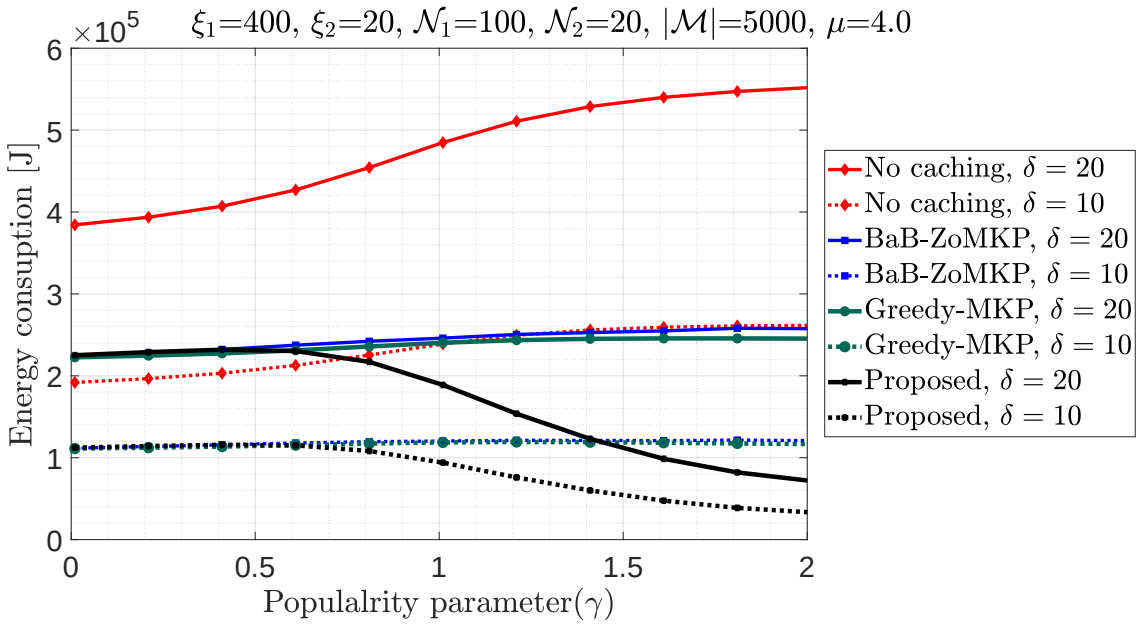

FIGURE 5.14: System utility cost for an increasing popularity index.

5.3.5 Section summary

In this section, we have studied the concept of cooperative joint caching, where both the placement and delivery processes are mutually optimized. In this case, the content placement targets optimizing the availability of contents while the content delivery process aims at reducing the service cost. It means that we have to consider the best possible decisions for the delivery process while optimizing the content placement, which gives a duality nature.

The utility cost is modeled based on how the content is delivered to active users and depends on parameters such as service distance, content size, UE density, and caching

size. Taking some numerical assumptions, critical parameters which should be verified by experimental data, for the MEC-computing tasks, we did simulations to analyze the performance of the proposed DJ-BOP strategy. Taking specific scenarios, we have shown that the proposed strategy outperforms the baseline system service cost reduction strategies. The proposed strategy is highly recommended in case of larger caching and smaller content sizes. More importantly, the DJ-BOP strategy dramatically reduces the utility cost when the content popularity index is higher.

5.4 Chapter Summary

In this chapter, we have studied cooperative content caching systems for MEC-enabled caching edges centrally controlled by the central head. The chapter had three sections that have various aspects of demand-aware content caching. The most important aspect of this chapter is that contents have different popularity towards different MHs, which makes the most comprehensive use cases of the HCN. We focused on assigning contents to the MHs where they are most 'demanded'.

In the first section, based on the request probability distribution, we modeled the caching scheme by the Separable Assignment Problems. This scenario is dedicated to the case where the non-partition concept applies. We then proposed an iterative combinatorial content selection algorithm, where an optimal solution is found for each MH. The outcomes of subproblems are merged such that the overlapping contents are fixed to only the edge where they have maximum demand. The obtained simulation results show that the proposed caching strategy outperforms profoundly. However, the performance alterations show that the network design is influenced by system parameters such as popular content library size, edges' cache size distribution, and content size and request probability distributions.

In the second section, we apply the caching strategy to multiple MHs that cooperatively cache contents by fragmenting. The level of content fragmenting depends on the application-level content synthesis and packet transmission protocols. The freedom to partition contents eases solving the demand-aware content caching by modeling with Special Generalized Knapsack Problems. Using simple relaxations, we assign 'top' chunks to MHs based on the weight of received request rates so that these top chunks can be served to requesting UE while other chunks are fetched from collaborating MHs. These top chunks are expected to increase the playback time and reduce jitter to improve the QoS. The more cooperative collaboration among the MHs and robust transmission protocol allows the partitioning of contents into the maximum possible number of chunks, which means when contents are fragmented to $|\mathcal{N}|$ chunks, it gives the best load balance and service reward. With the computational capacity of caching edges, the placement process can be updated when necessary. Such straightforward and very efficient optimization methods boost the performance of content caching.

Several use cases studied in this dissertation have room to extend into a more comprehensive network design perspective, such as the one modeled in Section 5.3. It aims to develop a more realistic caching approach where the content placement and

delivery processes are jointly optimized. This joint optimization is treated as a nested-constrained problem where the delivery decision is embedded into the placement decision. The content placement decision maximizes content availability, while the content delivery decision is made to minimize the delivery cost to end users. It addresses many constraints, such as the limited cache size, content non-partitioning, content non-repetition, different content popularity towards each MH, different content size, and preserving content demand for each MH. Numerical results show that the proposed DJ-BOP strategy reduces delivery costs by placing contents at the closest MH, which means the caching edge that demands the content higher than others.

Chapter 6

Conclusions and Future Prospects

The ever-increasing tsunami of content traffic in the HCN stems from the strong adoption of smart devices and the fast growth of high-bandwidth multimedia services. In addition, the high data consumption per subscriber to access resource-demanding services such as VoD, augmented reality, immersive media formats, and others continue to create a performance hurdle for the cellular network. To mitigate the pressing burden and meet the required QoE, this dissertation has focused on deeply studying existing works and proposing novel content caching frameworks.

This chapter summarizes the findings of the proposed caching strategies to enhance the performance of the HCN. The first section briefs our conclusions and main contributions, outlined in Section 1.4. The second section highlights future prospects, which had substantial progress during the research time.

6.1 Conclusions

Unlike the state-of-the-art, we have employed various combinatorial optimization techniques to uniquely model the content caching problems. Following implicit modeling for each scenario, appropriate optimal and exact algorithms are used to solve the formulated caching problem.

In **Chapter 3**, we study the problem of optimal content caching in cellular networks where MHs act as relays by caching video contents to offload MBSs. We design a novel caching strategy, DP-ZoSKP, to place contents from a single MBS to a single MH, with different content sizes and popularity. This caching problem is modeled by the 0/1-Knapsack Problems to optimally cache popular contents by taking the cache size and non-partitioning as constraints. Then, we delivered an optimal and enumerative caching strategy by introducing the tabulated dynamic programming method at a pseudo-polynomial time of complexity.

The DP-ZoSKP strategy outperforms baseline strategies for various network scenarios, considering CHP as an objective function. From the application-level simulation, we suggest useful design guidelines about the proposed strategy for real and heterogeneous network parameters, such as γ close to 1 and μ=1.23 Gb. The impact of these critical system parameters on CHP performance has been thoroughly investigated. The proposed strategy outperforms the baselines triple-fold in the case of contents having uniform popularity. In practical network deployment scenarios where

content popularity is Zipf distributed, and the content size is exponentially distributed, the DP-ZoSKP strategy outperforms up to 118% baselines. Both content popularity and size diversities, as part of the HCN heterogeneous property, are effectively managed by the proposed strategy. This work is a building block for other complex scenarios, investigated in consecutive chapters.

In **Chapter 4**, a more complex content caching strategy is investigated where multiple MEC-enabled MHs cache contents are uncoded. An exact caching strategy, BB-ZoMKP, is proposed using an advanced bound-and-bound algorithm that effectively handles the heterogeneity of network parameters. This caching system is comprehensive because multiple heterogeneous MHs are deployed, and contents have diversified sizes and defined popularity. We uniquely modeled the cooperative content placement problem using MKP formulation, subjected to practical constraints concerning the heterogeneity of cache sizes of cluster edges, the no-repetition of popular contents within the same cluster, and the no-partition of contents. To solve the proposed ZoMKP formulation, we propose an exact bound-and-bound search strategy. It employs a highly-effective methodology that explores the full state-space of the problem smartly. The most complex ZoMKP is reduced into several 0/1-Single Knapsack Problems (ZoSKP) formulations. A lower bound (LB) of the ZoMKP is obtained by sequentially fixing contents to each MH using the ZoSKP. On the other hand, an upper bound (UB) is obtained by assuming a cluster cache size equal to the sum of individual cache sizes of the MHs, by the standard surrogate relaxation. We then directly employ the ZoSKP formulation, whereas all ZoSKP formulations are solved using the tabular DP algorithm. Once contents are placed at the right edge, the hosting MH cooperates with a serving MH to serve the contents straightforwardly at minimized service cost.

Using extensive system-level simulations, we validate that the proposed bound-and-bound content placement strategy is efficient and computationally feasible under realistic MEC system setups. Its performance depends on the size distribution of contents and their popularity skewness. Several design guidelines and best practices are recommended through the simulation analysis, highlighting key performance trade-offs for MEC cluster formation and efficient content placement. The most important aspect of the proposed strategy is that the content selection algorithm can be scaled to any level of the HCN topology for predefined system parameters, such as content popularity predicted using any dynamic technique. We highlight the key performance trade-offs that govern the caching phases while we also derive a set of design guidelines and best practices for content placement in multi-tier MEC-enabled HCNs.

In the first section of **Chapter 5**, we analyze a caching strategy where content popularity varies towards each MH in a MEC cluster. This variation means each MH has different interest or demand level towards each content, typically the real HCN property. This placement problem is modeled using the ZoSAP by taking a series of constraints such as cache size, non-overlapping and non-partitioning of contents, and preserving the demand of each MH. We then provided a novel and efficient cooperative caching strategy using the recursive method at a very low cost, where each iteration is solved using the DP-ZoSKP algorithm. The outcomes of each subproblem are outfitted to get an instance where overlapping contents are fixed to maximally requesting MH. This

caching strategy, called D-ZoSAP, enables us to cache contents at MHs where they are most 'demanded' after the caching edges make RAN data analytics. For the first time in its kind, the strategy answers the two interdependent questions of caching: 'Where to cache a content?' and 'Which content to prioritize caching at an MH?'.

Our application-level analysis shows that the demand-aware D-ZoSAP caching strategy outperforms baseline caching strategies regarding content availability in the MEC cluster. Without loss of generality, the performance is influenced by critical system parameters such as library size, cache size distribution, and content size and popularity distributions. The demand-based placement places contents at MHs associated with heavily requesting UEs; thus, delivery cost overhead is minimized. In addition, the MEC functionality reduces the computation cost since the enabled MHs can perform data analytic tasks.

In the second section of **Chapter 5**, we give insight into the ZoMKP formulation but breaching the non-partition constraint. A powerful caching strategy is proposed by virtually pooling the available cluster cache size using relaxation techniques. The simplified form of the MKP is then solved using the DP-ZoSKP algorithm. In this scenario, contents can be freely partitioned into necessary chunks and placed at the target MH. Meanwhile, we propose the D-MKP partitioning strategy, where almost all contents are segmented into several chunks, whose attributes are determined in line with the level of content requests from corresponding MH. The D-MKP strategy gives a demand-based and fair content placement across MHs and excels the fronthaul load balancing at a very low computational cost. During content delivery, the MHs that host the chunks of the content cooperate to serve a sufficient number of chunks so that the UE can easily decode the content.

In the third section of **Chapter 5**, we heightened into a more comprehensive caching approach where the content placement and delivery are simultaneously optimized using a joint optimization scheme. This caching strategy ought to be the most complex scenario where several system constraints are considered, such as cache size, non-partitioning, non-repetition, different content popularity towards each MH, different content size, and preserving demand for each MH. It is overly complex because the content delivery problem is jointly embedded into the placement problem. In this setup, the content placement decision poses to maximize content availability, in terms of CHR, in the cluster while the content delivery decision stands to minimize the delivery cost to UEs. Until this time, we have modeled the content placement and delivery phases separately, with respective objectives. For the placement phase, we used the ZoSAP formulation to maximize the content availability (in terms of CHR) while for the delivery phase, we devised a cost model to minimize the end-to-end utility service cost. We then formulated a joint caching model called DJ-BOP, using bilevel optimization problems. Once the modeling is mathematically defined, it can be solved using either reduction methods or existing non-linear solver tools.

6.2 Future Prospects

The proposed caching strategies also inspire new research outlooks for the design of advanced MEC-enabled caching and communication schemes. In particular, we have been studying the following two caching prospects at different levels.

6.2.1 Caching with multipath protocols

Beyond the traditional protocols of UDP and TCP, there are newly developed multipath protocols such as multipath TCP (MPTCP) and multipath QUIC (MP-QUIC). Thus, multipath caching using these protocols is the anticipated technique where contents are cached into distributed network edges and served to users through defined multiple paths. This technique helps to simultaneously transport several content chunks during the placement and delivery phases. This approach executes extensive MEC tasks: contents are synthesized as frames at the source edge, frames scheduled to transmit through multiple paths, and frames are reordered at the receiver.

Given that the caching strategies use either a coded or uncoded form of content, as far as both caching cases deploy a content segmentation, the caching performance may not vary by using either way through multipath caching. However, uncoded caching is essential for the asynchronous transmission of frames in a distributed manner without increasing the communication rate [148]. Therefore, multipath caching conjoins the benefits of both coded and uncoded caching strategies.

Multipath caching is vital when delivering content to meet the expected QoE parameters, such as buffer, playback time, and service time. It also boosts the content placement process in case of occasional backhaul congestion. By exploiting the potential of the multipath protocols to access the content library via multiple routes, we can improve the network QoE at a reduced overhead compared to the cost of traditional caching. Again, the anticipated multipath caching is thought to merge caching schemes into the adaptive streaming technology with a small cache size demand. While applying the multipath technology, neither edge clustering nor coded caching is requisite, while demand-aware information-centric networking is very consequential for caching security and robustness.

As a big opportunity for multipath caching, from a protocols perspective, the expected 5G architecture is rich with several 3GPP and non-3GPP network types that can be used as servicing paths such as 3G UMTS, 4G LTE, 5G NR, WiFi, and MiFi. However, they have sizable heterogeneity in frequencies and standardization, which is an obstructive challenge. Fortunately, standardization entities are developing motivating techniques, such as dual connectivity, N3IWF, and ATSSS, to seamlessly integrate these technologies into one umbrella of the 5G NR [149].

Dual connectivity refers to the condition where one UE can simultaneously connect to multiple network interfaces and use radio resources without human interference, as detailed in the 3GPP technical specification (TS), Release 15 [150]. These devices use software interfaces to adapt the connectivity across broader generations. The dual

connectivity may operate in different frequencies from two base stations. Beyond establishing multipath, dual connectivity also helps switch a line-of-sight (LOS) path when a non-LOS occurs, a more severe problem in mmWave technology.

The N3IWF technology, recently detailed in Release 17 [151], is the technology that manages the interworking of untrusted-3GPP networks (basically the WLAN) with the 3GPP network (mainly the 5G core network) by transitioning the protocol data unite (PDU) sessions. The N3IWF helps as a gateway to use the 5GC through non-3GPP access or vice-versa. We may exploit the chance of creating multipath using multiple PDU sessions to support the ATSSS further and improve the QoE through network offloading, mobility engagement, and better coverage in dense areas.

The ATSSS functionality, introduced in Release 15 [152], is the technique that helps a UE to have simultaneous access to several 3GPP and non-3GPP networks, which might be trusted or untrusted-3GPP networks through multi-access PDU services. The *traffic steering* refers to the automatic selection of the best network type to use; the *switching* refers to the seamless handover from one network to the other, and *traffic splitting* refers to the network aggregation. These three sub-functionalities are used to decide on the active and inactive paths or best path characteristics and create load balances.

As explained above, these and other integration technologies open the way for implementing the futuristic multipath content caching, which will bring a paradigm shift in caching contents without placing the entire part. After deeply exploring disruptive multipath caching, we pioneered archiving a draft of an on-demand caching strategy. In an on-demand caching scheme, a preamble of the selected content is cached at edges, and its remaining portion is retrieved from the CS upon a request using the promising MP-QUIC protocol. However, choosing the integration technology and caching strategy need further investigation.

6.2.2 Demand-aware joint caching strategy

A novel joint and cooperative caching strategy, called DJ-BOP, is formulated in Subsection 5.3.5. In this scheme, the relationship between content caching and delivering processes is weakly interdependent. With different objective functions, since they are nested functions, they are jointly treated in a hierarchical formulation.

The placement problem without analytically optimizing the delivery phase could be easier, as it is already solved in Subsection 5.1. However, the BOP approach is very complex to solve optimally because it is lower-level constrained by delivery optimization. As a solution, this BOP is usually reformulated into mathematical programs with equilibrium constraints to solve it using existing non-linear solvers such as general algebraic modeling systems. Whereas there are several algorithms to solve it near-optimally, using reduction methods, we did not apply the actual bilevel optimization algorithms due to resource limitations. Instead, we heuristically analyzed the joint caching performance in terms of utility cost reduction. Therefore, future research is to devise an efficient and near-optimal algorithm to solve our BOP formulation. It also needs a consistent MEC task allocation technique, followed by an end-to-end numerical analysis and performance evaluation.

Bibliography

[1] M. Bande and V. V. Veeravalli, "Design of a heterogeneous cellular network with a wireless backhaul," *IEEE Transactions on Wireless Communications*, vol. 20, no. 1, pp. 243–253, 2021. DOI: 10.1109/TWC.2020.3024272.

[2] Ericsson, "Archive of mobility reports," June 2021. [Online]. Available: https://www.ericsson.com/en/reports-and-papers/mobility-report/reports.

[3] S. Safavat, N. N. Sapavath, and D. B. Rawat, "Recent advances in mobile edge computing and content caching," *Digital Communications and Networks*, vol. 6, no. 2, pp. 189–194, 2020, ISSN: 2352-8648. DOI: https://doi.org/10.1016/j.dcan.2019.08.004.

[4] Ericsson, "Mobility report," June 2021. [Online]. Available: https://www.ericsson.com/4a03c2/assets/local/mobility-report/documents/2021/june-2021-ericsson-mobility-report.pdf.

[5] X. He, K. Wang, H. Lu, W. Xu, and S. Guo, "Edge qoe: Intelligent big data caching via deep reinforcement learning," *IEEE Network*, vol. 34, no. 4, pp. 8–13, 2020. DOI: 10.1109/MNET.011.1900393.

[6] N. Golrezaei, A. F. Molisch, A. G. Dimakis, and G. Caire, "Femtocaching and device-to-device collaboration: A new architecture for wireless video distribution," *IEEE Communications Magazine*, vol. 51, no. 4, pp. 142–149, 2013. DOI: 10.1109/MCOM.2013.6495773.

[7] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile edge computing: A survey," *IEEE Internet Things J.*, vol. 5, no. 1, pp. 450–465, 2018. DOI: 10.1109/JIOT.2017.2750180.

[8] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing a key technology towards 5g," 2015. [Online]. Available: https://www.etsi.org/images/files/ETSIWhitePapers/etsi_wp11_mec_a_key_technology_towards_5g.pdf.

[9] A. Mehrabi, M. Siekkinen, and A. Ylä-Jaaski, "Qoe-traffic optimization through collaborative edge caching in adaptive mobile video streaming," *IEEE Access*, vol. 6, pp. 52 261–52 276, 2018. DOI: 10.1109/ACCESS.2018.2870855.

[10] Y. M. Saputra, D. T. Hoang, D. N. Nguyen, and E. Dutkiewicz, "A novel mobile edge network architecture with joint caching-delivering and horizontal cooperation," *IEEE Transactions on Mobile Computing*, vol. 20, no. 1, pp. 19–31, 2021. DOI: 10.1109/TMC.2019.2938510.

[11] S. Zhang, W. Sun, and J. Liu, "Spatially cooperative caching and optimization for heterogeneous network," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 11, pp. 11 260–11 270, 2019. DOI: 10.1109/TVT.2019.2941115.

[12] W. Jiang, G. Feng, and S. Qin, "Optimal cooperative content caching and delivery policy for heterogeneous cellular networks," *IEEE Transactions on Mobile Computing*, vol. 16, no. 5, pp. 1382–1393, 2017. DOI: 10.1109/TMC.2016.2597851.

[13] K. Shanmugam, N. Golrezaei, A. G. Dimakis, A. F. Molisch, and G. Caire, "Femtocaching: Wireless content delivery through distributed caching helpers," *IEEE Transactions on Information Theory*, vol. 59, no. 12, pp. 8402–8413, 2013. DOI: 10.1109/TIT.2013.2281606.

[14] S. Zhang, P. He, K. Suto, P. Yang, L. Zhao, and X. Shen, "Cooperative edge caching in user-centric clustered mobile networks," *IEEE Transactions on Mobile Computing*, vol. 17, no. 8, pp. 1791–1805, 2018. DOI: 10.1109/TMC.2017.2780834.

[15] A. Altieri, P. Piantanida, L. R. Vega, and C. G. Galarza, "On fundamental tradeoffs of device-to-device communications in large wireless networks," *IEEE Transactions on Wireless Communications*, vol. 14, no. 9, pp. 4958–4971, 2015. DOI: 10.1109/TWC.2015.2430341.

[16] C. Yang, Y. Yao, Z. Chen, and B. Xia, "Analysis on cache-enabled wireless heterogeneous networks," *IEEE Transactions on Wireless Communications*, vol. 15, no. 1, pp. 131–145, 2016. DOI: 10.1109/TWC.2015.2468220.

[17] N. Golrezaei, A. F. Molisch, A. G. Dimakis, and G. Caire, "Femtocaching and device-to-device collaboration: A new architecture for wireless video distribution," *IEEE Communications Magazine*, vol. 51, no. 4, pp. 142–149, 2013. DOI: 10.1109/MCOM.2013.6495773.

[18] L. Fan, Z. Dong, and P. Yuan, "The capacity of device-to-device communication underlaying cellular networks with relay links," *IEEE Access*, vol. 5, pp. 16 840–16 846, 2017. DOI: 10.1109/ACCESS.2017.2743778.

[19] Z. Chen, J. Lee, T. Q. S. Quek, and M. Kountouris, "Cooperative caching and transmission design in cluster-centric small cell networks," *IEEE Transactions on Wireless Communications*, vol. 16, no. 5, pp. 3401–3415, 2017. DOI: 10.1109/TWC.2017.2682240.

[20] Y. Wang, X. Tao, X. Zhang, and Y. Gu, "Cooperative caching placement in cache-enabled d2d underlaid cellular network," *IEEE Communications Letters*, vol. 21, no. 5, pp. 1151–1154, 2017. DOI: 10.1109/LCOMM.2017.2664805.

[21] N. Giatsoglou, K. Ntontin, E. Kartsakli, A. Antonopoulos, and C. Verikoukis, "D2d-aware device caching in mmwave-cellular networks," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 9, pp. 2025–2037, 2017. DOI: 10.1109/JSAC.2017.2720818.

[22] S. S. Kafıloğlu, G. Gür, and F. Alagöz, "Cooperative caching and video characteristics in d2d edge networks," *IEEE Communications Letters*, vol. 24, no. 11, pp. 2647–2651, 2020. DOI: 10.1109/LCOMM.2020.3009279.

[23] T. X. Tran, A. Hajisami, and D. Pompili, "Cooperative hierarchical caching in 5g cloud radio access networks," *IEEE Network*, vol. 31, no. 4, pp. 35–41, 2017. DOI: 10.1109/MNET.2017.1600307.

[24] Y. Cui and D. Jiang, "Analysis and optimization of caching and multicasting in large-scale cache-enabled heterogeneous wireless networks," *IEEE Transactions on Wireless Communications*, vol. 16, no. 1, pp. 250–264, 2017. DOI: 10.1109/TWC.2016.2622236.

[25] Y. Nam, S. Song, and J. Chung, "Clustered NFV service chaining optimization in mobile edge clouds," *IEEE Commun. Lett.*, vol. 21, no. 1, pp. 350–353, 2017. DOI: 10.1109/LCOMM.2016.2618788. [Online]. Available: https://doi.org/10.1109/LCOMM.2016.2618788.

[26] C. S. M. Babou, D. Fall, S. Kashihara, *et al.*, "Hierarchical load balancing and clustering technique for home edge computing," *IEEE Access*, vol. 8, pp. 127 593–127 607, 2020. DOI: 10.1109/ACCESS.2020.3007944. [Online]. Available: https://doi.org/10.1109/ACCESS.2020.3007944.

[27] W. You, C. Dong, X. Cheng, X. Zhu, Q. Wu, and G. Chen, "Joint optimization of area coverage and mobile-edge computing with clustering for fanets," *IEEE Internet Things J.*, vol. 8, no. 2, pp. 695–707, 2021. DOI: 10.1109/JIOT.2020.3006891. [Online]. Available: https://doi.org/10.1109/JIOT.2020.3006891.

[28] M. Bouet and V. Conan, "Mobile edge computing resources optimization: A geo-clustering approach," *IEEE Trans. Netw. Serv. Manag.*, vol. 15, no. 2, pp. 787–796, 2018. DOI: 10.1109/TNSM.2018.2816263. [Online]. Available: https://doi.org/10.1109/TNSM.2018.2816263.

[29] X. Song, Y. Geng, X. Meng, J. Liu, W. Lei, and Y. Wen, "Cache-enabled device to device networks with contention-based multimedia delivery," *IEEE Access*, vol. 5, pp. 3228–3239, 2017. DOI: 10.1109/ACCESS.2017.2664807.

[30] A. Sengupta, S. Amuru, R. Tandon, R. M. Buehrer, and T. C. Clancy, "Learning distributed caching strategies in small cell networks," in *2014* 11^{th} *International Symposium on Wireless Communications Systems (ISWCS)*, 2014, pp. 917–921. DOI: 10.1109/ISWCS.2014.6933484.

[31] T. Mihretu Ayenew, D. Xenakis, N. Passas, and L. Merakos, "A novel content placement strategy for heterogeneous cellular networks with small cells," *IEEE Networking Letters*, vol. 2, no. 1, pp. 10–13, 2020. DOI: 10.1109/LNET.2019.2950990.

[32] L. Li, G. Zhao, and R. S. Blum, "A survey of caching techniques in cellular networks: Research issues and challenges in content placement and delivery strategies," *IEEE Communications Surveys Tutorials*, vol. 20, no. 3, pp. 1710–1732, 2018. DOI: 10.1109/COMST.2018.2820021.

[33] J. Wen, K. Huang, S. Yang, and V. O. K. Li, "Cache-enabled heterogeneous cellular networks: Optimal tier-level content placement," *IEEE Transactions on Wireless Communications*, vol. 16, no. 9, pp. 5939–5952, 2017. DOI: 10.1109/TWC.2017.2717819.

[34] K. Poularakis, G. Iosifidis, A. Argyriou, I. Koutsopoulos, and L. Tassiulas, "Caching and operator cooperation policies for layered video content delivery," in *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*, 2016, pp. 1–9. DOI: 10.1109/INFOCOM.2016.7524427.

[35] A. Khreishah and J. Chakareski, "Collaborative caching for multicell-coordinated systems," in *2015 IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2015, pp. 257–262. DOI: 10.1109/INFCOMW.2015.7179394.

[36] R. Liu, H. Yin, X. Cai, *et al.*, "Cooperative caching scheme for content oriented networking," *IEEE Communications Letters*, vol. 17, no. 4, pp. 781–784, 2013. DOI: 10.1109/LCOMM.2013.020513.121680.

[37] K. S. Reddy and N. Karamchandani, "On the exact rate-memory trade-off for multi-access coded caching with uncoded placement," in *2018 International Conference on Signal Processing and Communications (SPCOM)*, 2018, pp. 1–5. DOI: 10.1109/SPCOM.2018.8724457.

[38] J. Kwak, Y. Kim, L. B. Le, and S. Chong, "Hybrid content caching in 5g wireless networks: Cloud versus edge caching," *IEEE Transactions on Wireless Communications*, vol. 17, no. 5, pp. 3030–3045, 2018. DOI: 10.1109/TWC.2018.2805893.

[39] S. A. R. Zaidi, M. Ghogho, and D. C. McLernon, "Information centric modeling for two-tier cache enabled cellular networks," in *2015 IEEE International Conference on Communication Workshop (ICCW)*, 2015, pp. 80–86. DOI: 10.1109/ICCW.2015.7247159.

[40] M. A. Kader, E. Bastug, M. Bennis, *et al.*, "Leveraging big data analytics for cache-enabled wireless networks," in *2015 IEEE Globecom Workshops (GC Wkshps)*, 2015, pp. 1–6. DOI: 10.1109/GLOCOMW.2015.7414014.

[41] E. Bastug, M. Bennis, and M. Debbah, "Living on the edge: The role of proactive caching in 5g wireless networks," *IEEE Commun. Mag.*, vol. 52, no. 8, pp. 82–89, 2014. DOI: 10.1109/MCOM.2014.6871674. [Online]. Available: https://doi.org/10.1109/MCOM.2014.6871674.

[42] T. Hou, G. Feng, S. Qin, and W. Jiang, "Proactive content caching by exploiting transfer learning for mobile edge computing," in *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, 2017, pp. 1–6. DOI: 10.1109/GLOCOM.2017.8254636.

[43] A. Ndikumana, N. H. Tran, D. H. Kim, K. T. Kim, and C. S. Hong, "Deep learning based caching for self-driving cars in multi-access edge computing," *IEEE Transactions on Intelligent Transportation Systems*, vol. 22, no. 5, pp. 2862–2877, 2021. DOI: 10.1109/TITS.2020.2976572.

[44] Z. Hu, Z. Zheng, T. Wang, L. Song, and X. Li, "Game theoretic approaches for wireless proactive caching," *IEEE Communications Magazine*, vol. 54, no. 8, pp. 37–43, 2016. DOI: 10.1109/MCOM.2016.7537175.

[45] Q. Xu, Z. Su, and R. Lu, "Game theory and reinforcement learning based secure edge caching in mobile social networks," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3415–3429, 2020. DOI: 10.1109/TIFS.2020.2980823.

[46] X. Li, X. Wang, K. Li, Z. Han, and V. C. M. Leung, "Collaborative multi-tier caching in heterogeneous networks: Modeling, analysis, and design," *IEEE Transactions on Wireless Communications*, vol. 16, no. 10, pp. 6926–6939, 2017. DOI: 10.1109/TWC.2017.2734646.

[47] S. Wang, X. Zhang, Y. Zhang, L. Wang, J. Yang, and W. Wang, "A survey on mobile edge networks: Convergence of computing, caching and communications," *IEEE Access*, vol. 5, pp. 6757–6779, 2017. DOI: 10.1109/ACCESS.2017.2685434.

[48] C. Hong and B. Varghese, "Resource management in fog/edge computing: A survey on architectures, infrastructure, and algorithms," *ACM Comput. Surv.*, vol. 52, no. 5, 97:1–97:37, 2019. DOI: 10.1145/3326066. [Online]. Available: https://doi.org/10.1145/3326066.

[49] J. Yao, T. Han, and N. Ansari, "On mobile edge caching," *IEEE Communications Surveys Tutorials*, vol. 21, no. 3, pp. 2525–2553, 2019. DOI: 10.1109/COMST.2019.2908280.

[50] S. Kumar, D. S. Vineeth, and A. F. A, "Edge assisted dash video caching mechanism for multi-access edge computing," in *2018 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS)*, 2018, pp. 1–6. DOI: 10.1109/ANTS.2018.8710106.

[51] T. Hou, G. Feng, S. Qin, and W. Jiang, "Proactive content caching by exploiting transfer learning for mobile edge computing," in *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, 2017, pp. 1–6. DOI: 10.1109/GLOCOM.2017.8254636.

[52] M. Fiedler, T. Hossfeld, and P. Tran-Gia, "A generic quantitative relationship between quality of experience and quality of service," *IEEE Network*, vol. 24, no. 2, pp. 36–41, 2010. DOI: 10.1109/MNET.2010.5430142.

[53] ITU, "Quality of service regulation manual," 2017. [Online]. Available: https://www.itu.int/dms_pub/itu-d/opb/pref/D-PREF-BB.QOS_REG01-2017-PDF-E.pdf.

[54] A. Lior, M. Jones, M. Arumaithurai, H. Tschofenig, and J. Korhonen, "Traffic classification and quality of service (qos) attributes for diameter," 2010. [Online]. Available: https://datatracker.ietf.org/doc/rfc5777/.

[55] Y. Chen, K. Wu, and Q. Zhang, "From qos to qoe: A tutorial on video quality assessment," *IEEE Communications Surveys Tutorials*, vol. 17, no. 2, pp. 1126–1165, 2015. DOI: 10.1109/COMST.2014.2363139.

[56] J. Nightingale, P. Salva-Garcia, J. M. A. Calero, and Q. Wang, "5g-qoe: Qoe modelling for ultra-hd video streaming in 5g networks," *IEEE Transactions on Broadcasting*, vol. 64, no. 2, pp. 621–634, 2018. DOI: 10.1109/TBC.2018.2816786.

[57] C. Li, L. Toni, J. Zou, H. Xiong, and P. Frossard, "Qoe-driven mobile edge caching placement for adaptive video streaming," *IEEE Transactions on Multimedia*, vol. 20, no. 4, pp. 965–984, 2018. DOI: 10.1109/TMM.2017.2757761.

[58] R. Coutinho, F. Chiariotti, D. Zucchetto, and A. Zanella, "Just-in-time proactive caching for dash video streaming," in *2018 17th Annual Mediterranean Ad Hoc Networking Workshop (Med-Hoc-Net)*, 2018, pp. 1–6. DOI: 10.23919/MedHocNet.2018.8407087.

[59] X. Chen, L. He, S. Xu, S. Hu, Q. Li, and G. Liu, "Hit ratio driven mobile edge caching scheme for video on demand services," in *2019 IEEE International Conference on Multimedia and Expo (ICME)*, 2019, pp. 1702–1707. DOI: 10.1109/ICME.2019.00293.

[60] H. Feng, Y. Jiang, D. Niyato, F.-C. Zheng, and X. You, "Content popularity prediction via deep learning in cache-enabled fog radio access networks," in *2019 IEEE Global Communications Conference (GLOBECOM)*, 2019, pp. 1–6. DOI: 10.1109/GLOBECOM38437.2019.9013376.

[61] Q. Wu, Z. Li, G. Tyson, S. Uhlig, M. A. Kaafar, and G. Xie, "Privacy-aware multipath video caching for content-centric networks," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 8, pp. 2219–2230, 2016. DOI: 10.1109/JSAC.2016.2577321.

[62] K. Poularakis, G. Iosifidis, V. Sourlas, and L. Tassiulas, "Exploiting caching and multicast for 5g wireless networks," *IEEE Transactions on Wireless Communications*, vol. 15, no. 4, pp. 2995–3007, 2016. DOI: 10.1109/TWC.2016.2514418.

[63] J. Cui, D. Wu, and Z. Qin, "Caching ap selection and channel allocation in wireless caching networks: A binary concurrent interference minimizing game solution," *IEEE Access*, vol. 6, pp. 54 516–54 526, 2018. DOI: 10.1109/ACCESS.2018.2871142.

[64] W. Jaafar, A. Mseddi, W. Ajib, and H. Elbiaze, "Content caching and channel allocation in d2d-assisted wireless hetnets," *IEEE Access*, vol. 9, pp. 112 502–112 515, 2021. DOI: 10.1109/ACCESS.2021.3103415.

[65] A. R. Elkordy, A. S. Motahari, M. Nafie, and D. Gündüz, "Cache-aided combination networks with interference," *IEEE Transactions on Wireless Communications*, vol. 19, no. 1, pp. 148–161, 2020. DOI: 10.1109/TWC.2019.2942913.

[66] T.-X. Zheng, H.-M. Wang, and J. Yuan, "Secure and energy-efficient transmissions in cache-enabled heterogeneous cellular networks: Performance analysis and optimization," *IEEE Transactions on Communications*, vol. 66, no. 11, pp. 5554–5567, 2018. DOI: 10.1109/TCOMM.2018.2873359.

[67] R. Pedarsani, M. A. Maddah-Ali, and U. Niesen, "Online coded caching," *IEEE/ACM Transactions on Networking*, vol. 24, no. 2, pp. 836–845, 2016. DOI: 10.1109/TNET.2015.2394482.

[68] K. Poularakis, G. Iosifidis, A. Argyriou, I. Koutsopoulos, and L. Tassiulas, "Distributed caching algorithms in the realm of layered video streaming," *IEEE Transactions on Mobile Computing*, vol. 18, no. 4, pp. 757–770, 2019. DOI: 10.1109/TMC.2018.2850818.

[69] M.-C. Lee and A. F. Molisch, "Caching policy and cooperation distance design for base station-assisted wireless d2d caching networks: Throughput and energy efficiency optimization and tradeoff," *IEEE Transactions on Wireless Communications*, vol. 17, no. 11, pp. 7500–7514, 2018. DOI: 10.1109/TWC.2018.2867596.

[70] N. Pappas, Z. Chen, and I. Dimitriou, "Throughput and delay analysis of wireless caching helper systems with random availability," *IEEE Access*, vol. 6, pp. 9667–9678, 2018. DOI: 10.1109/ACCESS.2018.2801246.

[71] M. Agiwal, A. Roy, and N. Saxena, "Next generation 5g wireless networks: A comprehensive survey," *IEEE Communications Surveys Tutorials*, vol. 18, no. 3, pp. 1617–1655, 2016. DOI: 10.1109/COMST.2016.2532458.

[72] ITU, "Setting the scene for 5g: Opportunities & challenges," 2018. [Online]. Available: https://www.itu.int/en/ITU-D/Documents/ITU_5G_REPORT-2018.pdf.

[73] H. Wang, R. Li, L. Fan, and H. Zhang, "Joint computation offloading and data caching with delay optimization in mobile-edge computing systems," in *2017 9th International Conference on Wireless Communications and Signal Processing (WCSP)*, 2017, pp. 1–6. DOI: 10.1109/WCSP.2017.8171178.

[74] L. Zhang, H.-C. Yang, and M. O. Hasna, "Generalized area spectral efficiency: An effective performance metric for green wireless communications," *IEEE Transactions on Communications*, vol. 62, no. 2, pp. 747–757, 2014. DOI: 10.1109/TCOMM.2013.122913.130138.

[75] D. Liu and C. Yang, "Caching policy toward maximal success probability and area spectral efficiency of cache-enabled hetnets," *IEEE Transactions on Communications*, vol. 65, no. 6, pp. 2699–2714, 2017. DOI: 10.1109/TCOMM.2017.2680447.

[76] D. Liu and C. Yang, "Energy efficiency of downlink networks with caching at base stations," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 4, pp. 907–922, 2016. DOI: 10.1109/JSAC.2016.2549398.

[77] Y. Duan, J. Zhang, W. Xia, and H. Zhu, "Energy efficiency of downlink c-ran with edge caching and fronthaul compression," *IEEE Communications Letters*, vol. 22, no. 12, pp. 2527–2530, 2018. DOI: 10.1109/LCOMM.2018.2873625.

[78] X. Zhang, T. Lv, W. Ni, J. M. Cioffi, N. C. Beaulieu, and Y. J. Guo, "Energy-efficient caching for scalable videos in heterogeneous networks," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 8, pp. 1802–1815, 2018. DOI: 10.1109/JSAC.2018.2844998.

[79] S. Nath, J. Wu, and J. Yang, "Optimum energy efficiency and age-of-information tradeoff in multicast scheduling," in *2018 IEEE International Conference on Communications (ICC)*, 2018, pp. 1–6. DOI: 10.1109/ICC.2018.8422521.

[80] Y. Lin, W. Bao, W. Yu, and B. Liang, "Optimizing user association and spectrum allocation in hetnets: A utility perspective," *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 6, pp. 1025–1039, 2015. DOI: 10.1109/JSAC.2015.2417011.

[81] A.-T. Tran, D. S. Lakew, T.-V. Nguyen, *et al.*, "Hit ratio and latency optimization for caching systems: A survey," in *2021 International Conference on Information Networking (ICOIN)*, 2021, pp. 577–581. DOI: 10.1109/ICOIN50884.2021.9334019.

[82] D. F. de Almeida, J. Yen, and M. Aibin, "Content delivery networks - q-learning approach for optimization of the network cost and the cache hit ratio," in *2020 IEEE Canadian Conference on Electrical and Computer Engineering (CCECE)*, 2020, pp. 1–5. DOI: 10.1109/CCECE47787.2020.9255813.

[83] L. Zhong, X. Zheng, Y. Liu, M. Wang, and Y. Cao, "Cache hit ratio maximization in device-to-device communications overlaying cellular networks," *China Communications*, vol. 17, no. 2, pp. 232–238, 2020. DOI: 10.23919/JCC.2020.02.018.

[84] H. Jin, D. Xu, C. Zhao, and D. Liang, "Information-centric mobile caching network frameworks and caching optimization: A survey," *EURASIP J. Wirel. Commun. Netw.*, vol. 2017, p. 33, 2017. DOI: 10.1186/s13638-017-0806-6.

[85] H. ElSawy, E. Hossain, and M. Haenggi, "Stochastic geometry for modeling, analysis, and design of multi-tier and cognitive cellular wireless networks: A survey," *IEEE Communications Surveys Tutorials*, vol. 15, no. 3, pp. 996–1019, 2013. DOI: 10.1109/SURV.2013.052213.00000.

[86] Y. Wang, W. Wang, Y. Cui, K. G. Shin, and Z. Zhang, "Distributed packet forwarding and caching based on stochastic network utility maximization," *IEEE/ACM Transactions on Networking*, vol. 26, no. 3, pp. 1264–1277, 2018. DOI: 10.1109/TNET.2018.2825460.

[87] V. S. Varma and T. Q. S. Quek, "Congestion games in caching enabled heterogeneous cellular networks," in *2015 IFIP Networking Conference (IFIP Networking)*, 2015, pp. 1–6. DOI: 10.1109/IFIPNetworking.2015.7145294.

[88] Z. Zhao, M. Peng, Z. Ding, W. Wang, and H. V. Poor, "Cluster content caching: An energy-efficient approach to improve quality of service in cloud radio access networks," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 5, pp. 1207–1221, 2016. DOI: 10.1109/JSAC.2016.2545384.

[89] Z. Su, Q. Xu, F. Hou, Q. Yang, and Q. Qi, "Edge caching for layered video contents in mobile social networks," *IEEE Transactions on Multimedia*, vol. 19, no. 10, pp. 2210–2221, 2017. DOI: 10.1109/TMM.2017.2733338.

[90] J. Wen, K. Huang, S. Yang, and V. O. K. Li, "Cache-enabled heterogeneous cellular networks: Optimal tier-level content placement," *IEEE Transactions on Wireless Communications*, vol. 16, no. 9, pp. 5939–5952, 2017. DOI: 10.1109/TWC.2017.2717819.

[91] O. Bello and S. Zeadally, "Intelligent device-to-device communication in the internet of things," *IEEE Syst. J.*, vol. 10, no. 3, pp. 1172–1182, 2016. DOI: 10.1109/JSYST.2014.2298837.

[92] S. O. Somuyiwa, A. György, and D. Gündüz, "A reinforcement-learning approach to proactive caching in wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 6, pp. 1331–1344, 2018. DOI: 10.1109/JSAC.2018.2844985.

[93] A. Sadeghi, F. Sheikholeslami, and G. B. Giannakis, "Optimal and scalable caching for 5g using reinforcement learning of space-time popularities," *IEEE J. Sel. Top. Signal Process.*, vol. 12, no. 1, pp. 180–190, 2018. DOI: 10.1109/JSTSP.2017.2787979.

[94] R. Singh, M. Srinivasan, and C. S. R. Murthy, "A learning based mobile user traffic characterization for efficient resource management in cellular networks," in *12th Annual IEEE Consumer Communications and Networking Conference, CCNC 2015, Las Vegas, NV, USA, January 9-12, 2015*, IEEE, 2015, pp. 304–309. DOI: 10.1109/CCNC.2015.7157993.

[95] K. Zhang, S. Leng, Y. He, S. Maharjan, and Y. Zhang, "Cooperative content caching in 5g networks with mobile edge computing," *IEEE Wireless Communications*, vol. 25, no. 3, pp. 80–87, 2018. DOI: 10.1109/MWC.2018.1700303.

[96] Y. Yu, "Mobile edge computing towards 5g: Vision, recent progress, and open challenges," *China Communications*, vol. 13, no. Supplement2, pp. 89–99, 2016. DOI: 10.1109/CC.2016.7833463.

[97] M. Garetto, E. Leonardi, and V. Martina, "A unified approach to the performance analysis of caching systems," *ACM Trans. Model. Perform. Evaluation Comput. Syst.*, vol. 1, no. 3, 12:1–12:28, 2016. DOI: 10.1145/2896380.

[98] H. Gomaa, G. G. Messier, C. Williamson, and R. Davies, "Estimating instantaneous cache hit ratio using markov chain analysis," *IEEE/ACM Transactions on Networking*, vol. 21, no. 5, pp. 1472–1483, 2013. DOI: 10.1109/TNET.2012.2227338.

[99] Z. Ming, M. Xu, and D. Wang, "Age-based cooperative caching in information-centric networking," in *2014 23rd International Conference on Computer Communication and Networks (ICCCN)*, 2014, pp. 1–8. DOI: 10.1109/ICCCN.2014.6911725.

[100] J. Liu, G. Wang, T. Huang, J. Chen, and Y. Liu, "Modeling the sojourn time of items for in-network cache based on lru policy," *China Communications*, vol. 11, no. 10, pp. 88–95, 2014. DOI: 10.1109/CC.2014.6969797.

[101] Q. D. Coninck and O. Bonaventure, "Multipath QUIC: design and evaluation," in *Proceedings of the 13th International Conference on emerging Networking EXperiments and Technologies, CoNEXT 2017, Incheon, Republic of Korea, December 12 - 15, 2017*, ACM, 2017, pp. 160–166. DOI: 10.1145/3143361.3143370.

[102] M. Amadeo, C. Campolo, G. Ruggeri, and A. Molinaro, "Beyond edge caching: Freshness and popularity aware iot data caching via ndn at internet-scale," *IEEE Transactions on Green Communications and Networking*, vol. 6, no. 1, pp. 352–364, 2022. DOI: 10.1109/TGCN.2021.3124452.

[103] J. Yao and N. Ansari, "Caching in dynamic iot networks by deep reinforcement learning," *IEEE Internet of Things Journal*, vol. 8, no. 5, pp. 3268–3275, 2021. DOI: 10.1109/JIOT.2020.3004394.

[104] S. Zhang and J. Liu, "Optimal probabilistic caching in heterogeneous iot networks," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 3404–3414, 2020. DOI: 10.1109/JIOT.2020.2969466.

[105] S. Vural, P. Navaratnam, N. Wang, C. Wang, L. Dong, and R. Tafazolli, "In-network caching of internet-of-things data," in *IEEE International Conference on Communications, ICC 2014, Sydney, Australia, June 10-14, 2014*, IEEE, 2014, pp. 3185–3190. DOI: 10.1109/ICC.2014.6883811.

[106] M. Meddeb, A. Dhraief, A. Belghith, T. Monteil, and K. Drira, "How to cache in icn-based iot environments?" In *14th IEEE/ACS International Conference on Computer Systems and Applications, AICCSA 2017, Hammamet, Tunisia, October 30 - Nov. 3, 2017*, IEEE Computer Society, 2017, pp. 1117–1124. DOI: 10.1109/AICCSA.2017.37.

[107] Z. Su, Y. Hui, Q. Xu, T. Yang, J. Liu, and Y. Jia, "An edge caching scheme to distribute content in vehicular networks," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 6, pp. 5346–5356, 2018. DOI: 10.1109/TVT.2018.2824345.

[108] C. Chen, C. Wang, T. Qiu, M. Atiquzzaman, and D. O. Wu, "Caching in vehicular named data networking: Architecture, schemes and future directions," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 4, pp. 2378–2407, 2020. DOI: 10.1109/COMST.2020.3005361.

[109] G. Zhong, J. Yan, and L. Kuang, "Qoe-driven social aware caching placement for terrestrial-satellite networks," *China Communications*, vol. 15, no. 10, pp. 60–72, 2018. DOI: 10.1109/CC.2018.8485469.

[110] X. Zhu, C. Jiang, L. Kuang, and Z. Zhao, "Cooperative multilayer edge caching in integrated satellite-terrestrial networks," *IEEE Transactions on Wireless Communications*, vol. 21, no. 5, pp. 2924–2937, 2022. DOI: 10.1109/TWC.2021.3117026.

[111] E. Wang, H. Li, and S. Zhang, "Load balancing based on cache resource allocation in satellite networks," *IEEE Access*, vol. 7, pp. 56 864–56 879, 2019. DOI: 10.1109/ACCESS.2019.2914167.

[112] X. Zhang, B. Zhang, K. An, G. Zheng, S. Chatzinotas, and D. Guo, "Stochastic geometry-based analysis of cache-enabled hybrid satellite-aerial-terrestrial networks with non-orthogonal multiple access," *IEEE Transactions on Wireless Communications*, vol. 21, no. 2, pp. 1272–1287, 2022. DOI: 10.1109/TWC.2021.3103499.

[113] S. Soleimani, X. Tao, and Y. Li, "Cooperative group caching strategy in content-centric wireless ad hoc networks," in *2018 IEEE/CIC International Conference on Communications in China (ICCC)*, 2018, pp. 793–797. DOI: 10.1109/ICCChina.2018.8641166.

[114] T. Zhang, X. Xu, L. Zhou, X. Jiang, and J. Loo, "Cache space efficient caching scheme for content-centric mobile ad hoc networks," *IEEE Systems Journal*, vol. 13, no. 1, pp. 530–541, 2019. DOI: 10.1109/JSYST.2018.2851394.

[115] N. Carlsson and D. Eager, "Ephemeral content popularity at the edge and implications for on-demand caching," *IEEE Transactions on Parallel and Distributed Systems*, vol. 28, no. 6, pp. 1621–1634, 2017. DOI: 10.1109/TPDS.2016.2614805.

[116] X. Ge, J. Ye, Y. Yang, and Q. Li, "User mobility evaluation for 5g small cell networks based on individual mobility model," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 3, pp. 528–541, 2016. DOI: 10.1109/JSAC.2016.2525439.

[117] H. ElSawy, A. Sultan-Salem, M.-S. Alouini, and M. Z. Win, "Modeling and analysis of cellular networks using stochastic geometry: A tutorial," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 1, pp. 167–203, 2017. DOI: 10.1109/COMST.2016.2624939.

[118] J. Riihijarvi and P. Mahonen, "Machine learning for performance prediction in mobile cellular networks," *IEEE Computational Intelligence Magazine*, vol. 13, no. 1, pp. 51–60, 2018. DOI: 10.1109/MCI.2017.2773824.

[119] L. N. T. Huynh, Q.-V. Pham, T. D. T. Nguyen, M. D. Hossain, Y.-R. Shin, and E.-N. Huh, "Joint computational offloading and data-content caching in noma-mec networks," *IEEE Access*, vol. 9, pp. 12 943–12 954, 2021. DOI: 10.1109/ACCESS.2021.3051278.

[120] M. Roddy, T. Truong, P. Walsh, *et al.*, "5g network slicing for mission-critical use cases," in *2019 IEEE 2nd 5G World Forum (5GWF)*, 2019, pp. 409–414. DOI: 10.1109/5GWF.2019.8911651.

[121] A. Pietrabissa, F. D. Priscoli, A. D. Giorgio, A. Giuseppi, M. Panfili, and V. Suraci, "An approximate dynamic programming approach to resource management in multi-cloud scenarios," *Int. J. Control*, vol. 90, no. 3, pp. 492–503, 2017. DOI: 10.1080/00207179.2016.1185802. [Online]. Available: https://doi.org/10.1080/00207179.2016.1185802.

[122] M. Moghimi, A. Zakeri, M. R. Javan, N. Mokari, and D. W. K. Ng, "Joint radio resource allocation and cooperative caching in pd-noma-based hetnets," *IEEE Transactions on Mobile Computing*, vol. 21, no. 6, pp. 2029–2044, 2022. DOI: 10.1109/TMC.2020.3034618.

[123] YouTube, "Recommended upload encoding settings," 2021. [Online]. Available: https://support.google.com/youtube/answer/1722171?hl=en.

[124] NTT-DOCOMO, "Guidelines for video delivery over a mobile network-version 1.0," 2014. [Online]. Available: https://www.nttdocomo.co.jp/english/binary/pdf/service/developer/smart_phone/technical_info/etc/mobile_movie_guide_1_0_En.pdf.

[125] X. Che, B. Ip, and L. Lin, "A survey of current youtube video characteristics," *IEEE MultiMedia*, vol. 22, no. 2, pp. 56–63, 2015. DOI: 10.1109/MMUL.2015.34.

[126] I. T. U. (ITU), "Setting the scene for 5g: Opportunities & challenges," 2020. [Online]. Available: https://www.itu.int/en/ITU-D/Documents/ITU_5G_REPORT-2018.pdf.

[127] M. A. Salahuddin, J. Sahoo, R. Glitho, H. Elbiaze, and W. Ajib, "A survey on content placement algorithms for cloud-based content delivery networks," *IEEE Access*, vol. 6, pp. 91–114, 2018. DOI: 10.1109/ACCESS.2017.2754419.

[128] H. S. Goian, O. Y. Al-Jarrah, S. Muhaidat, Y. Al-Hammadi, P. Yoo, and M. Dianati, "Popularity-based video caching techniques for cache-enabled networks: A survey," *IEEE Access*, vol. 7, pp. 27 699–27 719, 2019. DOI: 10.1109/ACCESS.2019.2898734.

[129] ETSI, "Mec in 5g networks, etsi white paper no.28," 2018. [Online]. Available: https://www.etsi.org/images/files/ETSIWhitePapers/etsi_wp28_mec_in_5G_FINAL.pdf.

[130] C. Li, L. Toni, J. Zou, H. Xiong, and P. Frossard, "Qoe-driven mobile edge caching placement for adaptive video streaming," *IEEE Transactions on Multimedia*, vol. 20, no. 4, pp. 965–984, 2018. DOI: 10.1109/TMM.2017.2757761.

[131] T. X. Tran, A. Hajisami, and D. Pompili, "Cooperative hierarchical caching in 5g cloud radio access networks," *IEEE Network*, vol. 31, no. 4, pp. 35–41, 2017. DOI: 10.1109/MNET.2017.1600307.

[132] S. Martello and P. Toth, "A bound and bound algorithm for the zero-one multiple knapsack problem," *Discret. Appl. Math.*, vol. 3, no. 4, pp. 275–288, 1981. DOI: 10.1016/0166-218X(81)90005-6. [Online]. Available: https://doi.org/10.1016/0166-218X(81)90005-6.

[133] P. H. Vance, "Knapsack problems: Algorithms and computer implementations (S. martello and p. toth)," *SIAM Rev.*, vol. 35, no. 4, pp. 684–685, 1993. DOI: 10.1137/1035174.

[134] A.-T. Tran, N.-N. Dao, and S. Cho, "Bitrate adaptation for video streaming services in edge caching systems," *IEEE Access*, vol. 8, pp. 135 844–135 852, 2020. DOI: 10.1109/ACCESS.2020.3011517.

[135] X. Zhang, T. Lv, W. Ni, J. M. Cioffi, N. C. Beaulieu, and Y. J. Guo, "Energy-efficient caching for scalable videos in heterogeneous networks," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 8, pp. 1802–1815, 2018. DOI: 10.1109/JSAC.2018.2844998.

[136] B. N. Bharath, K. G. Nagananda, and H. V. Poor, "A learning-based approach to caching in heterogenous small cell networks," *IEEE Transactions on Communications*, vol. 64, no. 4, pp. 1674–1686, 2016. DOI: 10.1109/TCOMM.2016.2536728.

[137] X. Zhong, X. Wang, L. Li, *et al.*, "CL-ADMM: A cooperative-learning-based optimization framework for resource management in MEC," *IEEE Internet Things J.*, vol. 8, no. 10, pp. 8191–8209, 2021. DOI: 10.1109/JIOT.2020.3043749. [Online]. Available: https://doi.org/10.1109/JIOT.2020.3043749.

[138] D. T. Hoang, D. Niyato, D. N. Nguyen, E. Dutkiewicz, P. Wang, and Z. Han, "A dynamic edge caching framework for mobile 5g networks," *IEEE Wireless Communications*, vol. 25, no. 5, pp. 95–103, 2018. DOI: 10.1109/MWC.2018.1700360.

[139] N. Giatsoglou, K. Ntontin, E. Kartsakli, A. Antonopoulos, and C. Verikoukis, "D2d-aware device caching in mmwave-cellular networks," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 9, pp. 2025–2037, 2017. DOI: 10.1109/JSAC.2017.2720818.

[140] L. N. T. Huynh, Q. Pham, T. D. T. Nguyen, M. D. Hossain, Y. Shin, and E. Huh, "Joint computational offloading and data-content caching in NOMA-MEC networks," *IEEE Access*, vol. 9, pp. 12 943–12 954, 2021.

[141] T. M. Ayenew, D. Xenakis, N. Passas, and L. Merakos, "Cooperative content caching in mec-enabled heterogeneous cellular networks," *IEEE Access*, vol. 9, pp. 98 883–98 903, 2021. DOI: 10.1109/ACCESS.2021.3095356.

[142] L. Fleischer, M. X. Goemans, V. S. Mirrokni, and M. Sviridenko, "Tight approximation algorithms for maximum separable assignment problems," *Math. Oper. Res.*, vol. 36, no. 3, pp. 416–431, 2011. DOI: 10.1287/moor.1110.0499. [Online]. Available: https://doi.org/10.1287/moor.1110.0499.

[143] N. K. Panigrahy, J. Li, and D. Towsley, "Hit rate vs. hit probability based cache utility maximization," *SIGMETRICS Perform. Eval. Rev.*, vol. 45, no. 2, 21–23, 2017, ISSN: 0163-5999. DOI: 10.1145/3152042.3152050.

[144] D. Zhao, Z. Yan, M. Wang, P. Zhang, and B. Song, "Is 5g handover secure and private? a survey," *IEEE Internet of Things Journal*, vol. 8, no. 16, pp. 12 855–12 879, 2021. DOI: 10.1109/JIOT.2021.3068463.

[145] B. Yang, X. Yang, X. Ge, and Q. Li, "Coverage and handover analysis of ultra-dense millimeter-wave networks with control and user plane separation architecture," *IEEE Access*, vol. 6, pp. 54 739–54 750, 2018. DOI: 10.1109/ACCESS.2018.2871363.

[146] M. Yan, C. A. Chan, W. Li, L. Lei, A. F. Gygax, and C.-L. I, "Assessing the energy consumption of proactive mobile edge caching in wireless networks," *IEEE Access*, vol. 7, pp. 104 394–104 404, 2019. DOI: 10.1109/ACCESS.2019.2931449.

[147] A. Vishwanath, F. Jalali, K. Hinton, T. Alpcan, R. W. A. Ayre, and R. S. Tucker, "Energy consumption comparison of interactive cloud-based and local applications," *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 4, pp. 616–626, 2015. DOI: 10.1109/JSAC.2015.2393431.

[148] K. S. Reddy and N. Karamchandani, "On the exact rate-memory trade-off for multi-access coded caching with uncoded placement," in *2018 International Conference on Signal Processing and Communications (SPCOM), Bangalore, India, July 16-19, 2018*, IEEE, 2018, pp. 1–5. DOI: 10.1109/SPCOM.2018.8724457.

[149] V. Agarwal, C. Sharma, R. Shetty, A. Jangam, and R. Asati, "A journey towards a converged 5g architecture & beyond," in *2021 IEEE 4th 5G World Forum (5GWF)*, 2021, pp. 18–23. DOI: 10.1109/5GWF52925.2021.00011.

[150] 3GPP, "Technical Specification Group Services and System Aspects," 3rd Generation Partnership Project (3GPP), Technical Specification (TS) 21.915, Jan. 2019, Version 15.0.0. [Online]. Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3389.

[151] 3GPP, "Security Assurance Specification for Non-3GPP InterWorking Function (N3IWF)," 3rd Generation Partnership Project (3GPP), Technical Specification (TS) 33.520, Aug. 2021, Version 17.3.0. [Online]. Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3748.

[152] 3GPP, "Access Traffic Steering, Switching and Splitting (ATSSS)," 3rd Generation Partnership Project (3GPP), Technical Specification (TS) 24.193, Sep. 2019, Version 16.4.0. [Online]. Available: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=3607.