Alma Mater Studiorum – Università di Bologna

DOTTORATO DI RICERCA IN

INGEGNERIA ELETTRONICA, TELECOMUNICAZIONI E TECNOLOGIE DELL'INFORMAZIONE

Ciclo 33

Settore Concorsuale: 09/F2 - TELECOMUNICAZIONI

Settore Scientifico Disciplinare: ING-INF/03 - TELECOMUNICAZIONI

LIFE-CYCLE MANAGEMENT AND PLACEMENT OF SERVICE FUNCTION CHAINS IN MEC-ENABLED 5G NETWORKS

Presentata da: Rasoul Behravesh

Coordinatore Dottorato

Prof. Alessandra Costanzo

Supervisore

Dr. Davit Harutyunyan

Co-supervisore

Prof. Walter Cerroni

Dr. Roberto Riggio

Esame finale anno 2021

Declaration of Authorship

I, RASOUL BEHRAVESH, declare that this thesis titled, "Life-Cycle Management and Placement of Service Function Chains in MEC-Enabled 5G 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 thesis 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 consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Rasoul Behravesh
April 2021

If someday a delegate comes to my land
And asks me:
"Where is the grave of the Unknown Soldier here?"
I will tell him:
"Sir,
On the bank of any stream,
On the bench of any mosque,
In the shade of any home,
On the threshold of any church,
At the mouth of any cave,
In the mountains on any rock,
In the gardens on any tree,
In my country,
On any span of land,
Under any cloud in the sky,
Do not worry,
Make a slight bow,
And place your wreath of flowers."

Abdullah Pashew, Kurdish poet
The Unknown Soldier

Acknowledgements

Accomplishing a Ph.D. thesis is not an individual experience; rather, it takes place in a social context, including collaborations of a team - my thanks and appreciation to all of them for being part of this journey and making this thesis possible.

First of all, I wish to express my deepest gratitude to my Ph.D. advisor **Dr. Davit Harutyunyan** for his indispensable help to shape my research topic, continuous support, and advice during the Ph.D. journey. Without his enthusiasm, encouragement, support, and constant optimism, this thesis would hardly have been completed. **Davit** was a helpful advisor and a true friend who was always available to discuss and give suggestions. I would also like to extend my gratitude to my co-advisors **Prof. Walter Cerroni** and **Dr. Roberto Riggio** for their support during my Ph.D. degree.

During my Ph.D. I spent six months as a visiting researcher at the RISE Research Institute of Sweden, under the supervision of **Dr. Rebecca Steinert**. I consider myself fortunate to have had **Rebecca** as my advisor during the visiting period, where she gave me the freedom to choose my research topic and continuously supported me during the whole period. I learned a lot from her, and I am so thankful that she gave me the opportunity to work and collaborate with her team. Briefly, **Rebecca** is a fantastic manager that everyone wishes to work with.

A special thanks to my co-authors **Daniel F. Perez-Ramirez** and **Akhila Rao** without whom finishing this thesis would be impossible.

To my friends **Behrouz** and **Peyman**, you should know that your support and encouragement was worth more than I can express on paper. You know it was a long and sometimes bumpy road, but you reassured and supported me along the way. Special thanks go to my friend **Alina Karakanta** for proof-reading this dissertation and providing constructive criticisms.

Finally, my deep and sincere gratitude to my family for their continuous and unparalleled love, help, and support. I am forever indebted to my parents **Qaraman** and **Mahboube**, for giving me the opportunities and experiences that have made me who I am. I am grateful to my sister **Chiman** (including her sweet son **Bardiya**) and my brother **Zanyar** for always being there for me as a friend. They selflessly encouraged me to explore new directions in life and seek my own destiny. This journey would not have been possible if not for them, and I dedicate this milestone to them. I would also like to send my love and appreciation to my grandfather **Osman**, whom I am sure would be proud of me if he was alive. I would also like to acknowledge my grandmother **Manije** and my uncles **Kave**, **Keyvan**, **Nasrin**, **Akam**, and **Aso** for their unconditional love and support during whole my life. You are truly amazing, and you have my utmost love.

Abstract

Recent advancements in mobile communication technology have led to the fifth generation of mobile cellular networks (5G), driven by the proliferation in data traffic demand, stringent latency requirements, and the desire for a fully connected world. The adoption of the 5G technology is becoming a necessity for Mobile Network Operators (MNOs) to remain competitive in the market and efficiently cope with the stringent requirements in terms of latency, data rate, coverage, and providing support for many futuristic use cases and services. This transformation calls for novel technology solutions such as Multi-Access Edge Computing (MEC) and Network Function Virtualization (NFV) to satisfy service requirements while providing dynamicity and instant service deployment for the users.

MEC and NFV are two principal and complementary enablers for 5G networks whose co-existence can lead to numerous benefits. While the former intends to provide cloud computing capabilities at the edge of the network, low-latency services to the end-users, real-time access to Radio Access Network (RAN) information, and offload the backhaul, the latter exploits the potential of virtualization technology to decouple Network Functions (NFs) such as firewalls, Intrusion Detection System (IDS), and Intrusion Prevention System (IPS) from vendor-specific and dedicated servers and enabling them to run on top of standard servers as Virtual Network Functions (VNFs).

Despite the numerous advantages MEC offers, physical resources at the edge are extremely scarce and require efficient utilization. Moreover, the heterogeneity of these resources further necessitates devising generic methods applicable to different platforms. Finally, considering the roll-out of 5G networks, a higher number of users join/leave the network or update their service requirements. Consequently, novel approaches are needed to adapt the network to these changes in order to avoid resource over-utilization while meeting user requirements.

In this doctoral dissertation, we first attempt to optimize resource utilization (mainly storage) at the network edge for the scenario of live video streaming. We

specifically consider the problem of DASH video prefetching and studying the trade-off between different video prefetching/caching options with the goal of maximizing the number of users served from the edge. The work aims to utilize the real-time RAN information available at the MEC servers to develop a Machine Learning (ML)-based prediction solution and anticipate user requests. Consequently, prefetching approaches are used to prefetch/cache video contents from a centralized video server. We then carry out a relative comparison between proposed prefetching approaches to verify the applicability of the proposed solutions in different network configurations.

Regarding the advantages of NFV technology for the deployment of Network Functions (NFs), the second problem that this dissertation address is the proper association of the users to the gNBs (base station in 5G networks) along with efficient placement of Service Function Chains (SFCs) on the substrate network. We consider a 5G network, enabled with the MEC technology that can be used to host applications as well NFs deployed as VNFs. Our primary purpose is to find a proper embedding of the SFCs in a hierarchical 5G network. The problem is formulated as a Mixed Integer Linear Programming (MILP) model, having the objective to minimize service provisioning cost, link utilization, and the effect of VNF migration on users' perceived Quality of Experience (QoE). A heuristic algorithm is also proposed, following the objective of minimizing the number of users affected by VNF migration. The proposed algorithms provide MNOs with various options to select between promptness, solution optimality, and user satisfaction.

After rigorously analyzing the proposed SFC placement and considering the dynamicity of mobile networks, our next objective is to develop an approach that can adjust the network based on the users' varying demands. Therefore, we develop an Integer Linear Programming (ILP) based model that aims to minimize the resource provisioning cost by dynamically embed and scale SFCs so that provisioning cost is minimized while user requirements are met. Specifically, we consider different VNF scaling strategies, including vertical, horizontal, and hybrid, with a particular emphasis on studying the trade-offs between the vertical and horizontal VNF scaling strategies. The time complexity of the ILP model is tackled by proposing a heuristic algorithm, which performs comparably to the ILP hybrid approach.

Keywords. 5G, MEC, NFV, VNE, ILP, MILP, SFC Placement, Resource Allocation, User Association, DASH, Scaling, Machine Learning.

Contents

List of Figures

List of Tables

List of Abbreviations

3GPP	**3rd G**eneration **P**artnership **P**roject
4G	**4th G**eneration of Mobile Cellular Networks
5G	**5th G**eneration of Mobile Cellular Networks
ABR	**A**daptive **B**itrate
AF	**A**pplication **F**unction
AI	**A**rtificial **I**ntelligence
AMF	**A**ccess and **M**obility **M**anagement **F**unction
AP	**A**ccess **P**oint
API	**A**pplication **P**rogramming **I**nterface
AR	**A**ugmented **R**eality
BB	**B**ranch & **B**ound
BC	**B**ranch & **C**ut
BS	**B**ase **S**tation
CapEx	**Cap**ital **Ex**penditure
CDN	**C**ontent **D**elivery **N**etwork
COTS	**C**ommercial-**O**ff-**T**he-**S**helf
CPF	**C**ontrol **P**lane **F**unction
CQI	**C**hannel **Q**uality **I**ndicator
C-RAN	**C**loud-**R**adio **A**ccess **N**etwork
CSI	**C**hannel **S**tate **I**nformation
CUPS	**C**ontrol and **D**ata **P**lane **S**eparation
DASH	**D**ynamic **A**daptive **S**treaming over **H**TTP
DC	**D**ata **C**enter
eMBB	**e**nhanced **M**obile **B**roadband
EMS	**E**lement **M**anagement **S**ystem
ETSI	**E**uropean **T**elecommunication **S**tandards **I**nstitute
E2E	**E**nd-**to**-**E**nd
gNB	**g**Node**B**
HAS	**H**TTP **A**daptive **S**treaming
IDS	**I**ntrusion **D**etection **S**ystem
ILP	**I**nteger **L**inear **P**rogramming

IMT	**I**nternational **M**obile **T**elecommunications
IoT	**I**nternet of **T**hing
IPS	**I**ntrusion **P**revention **S**ystem
ISG	**I**ndustry **S**pecification **G**roup
ISO	**I**nternational **O**rganization for **S**tandardization
IT	**I**nformation **T**echnology
ITU	**I**nternational **T**elecommunication **U**nion
ITU-T	**ITU** - **T**elecommunication Standardization Sector
KPI	**K**ey **P**erformance **I**ndicator
LCM	**L**ife-**C**ycle **M**anagement
LRU	**L**east **R**ecently **U**sed
LTE	**L**ong **T**erm **E**volution
MANO	**MAN**agement and **O**rchestration
MCS	**M**odulation and **C**oding **S**cheme
MEC	**M**ulti-access **E**dge **C**omputing
MILP	**M**ixed **I**nteger **L**inear **P**rogramming
ML	**M**achine **L**earning
MIMO	**M**ultiple **I**nput **M**ultiple **O**utput
mMTC	**m**assive **M**achine **T**ype **C**ommunication
mmWave	**M**illimeter **Waves**
MNO	**M**obile **N**etwork **O**perator
MPD	**M**edia **P**resentation **D**escription
MPEG	**M**oving **P**icture **E**xperts **G**roup
MVNO	**M**obile **V**irtual **N**etwork **O**perator
NF	**N**etwork **F**unction
NFV	**N**etwork **F**unction **V**irtualization
NFVI	**NFV I**nfrastructure
NRF	**N**etwork **R**epository **F**unction
NS	**N**etwork **S**licing
NSA	**N**one **S**tand **A**lone
OFDM	**O**rthogonal **F**requency **D**ivision **M**ultiplexing
OpEx	**Op**erational **Ex**penditure
OS	**O**perating **S**ystem
OTT	**O**ver **T**he **T**op
POC	**P**roof **O**f **C**oncept
QoE	**Q**uality **o**f **E**xperience
QoS	**Q**uality **o**f **S**ervice
RAN	**R**adio **A**ccess **N**etwork

RF	**R**andom **F**orest
RNIS	**R**adio **N**etwork **I**nformation **S**ervice
RSSI	**R**eceived **S**ignal **S**trength **I**ndicator
SBA	**S**ervice **B**ased **A**rchitecture
SDN	**S**oftware **D**efined **N**etworking
SFC	**S**ervice **F**unction **C**hain
SINR	**S**ignal to **N**oise and **I**nterference **R**atio
SLA	**S**ervice **L**evel **A**greement
SMF	**S**ession **M**anagement **F**unction
STF	**S**tateful **F**unction
TTI	**T**ransmission **T**ime **I**nterval
URLLC	**U**ltra-**R**elaiable **L**ow-**L**atency **C**ommunication
UDR	**U**nified **D**ata **R**epository
UE	**U**ser **E**quipment
UPF	**U**ser **P**lane **F**unction
USDF	**U**nstructured **D**ata **S**torage **F**unction
VIM	**V**irtualized **I**nfrastructure **M**anager
VM	**V**irtual **M**achine
VNE	**V**irtual **N**etwork **E**mbedding
VNF	**V**irtual **N**etwork **F**unction
VNR	**V**irtual **N**etwork **R**equests
VR	**V**irtual **R**eality
XGB	**G**radient **B**oosting Trees

I dedicate this thesis to my parents. . .

Chapter 1

Introduction

This chapter outlines the motivations and objectives of this dissertation, states the problems, the methodologies employed to solve the problems, and draws the main results. First, we describe the necessities of the newborn applications such as Virtual Reality (VR), Augmented Reality (AR), autonomous cars, Internet of Things (IoT), and the challenges towards the implementation of the fifth generation of mobile networks (5G). Second, we introduce the leading enabler technologies for 5G to fulfill the requirements of these applications. Third, we summarize the optimization approaches undertaken to tackle the problem of lifecycle management and placement of SFCs in the context of 5G networks endowed with the MEC technology. Finally, we draw the structure of the dissertation and preview the contents of the subsequent chapters.

1.1 Motivations and Objectives

Technological development can significantly contribute to human lives to make it more comfortable, enjoyable, safer, and healthier. Lately, human-technology interaction has been rapidly growing and profoundly transforming the way humans interact with their outside world. Recent communication developments have made smart devices an integral part of human lives [1]. Consequently, a mushrooming number of internet-capable devices and services communicate through the network and generate an unprecedented amount of traffic. Cisco forecasts that by 2022 the number of mobile-connected devices per capita will reach 1.5, and as a consequence, the annual generated traffic will reach almost one zettabyte in the same year [2]. Thus, on the one hand, mobile data traffic has been growing immensely, forcing Mobile

Network Operators (MNOs) to increase the network capacity to accommodate the traffic demand. On the other hand, the current mobile networks are not flexible and cannot react based on traffic change in the network.

The new emerging applications and the excessive number of internet-capable devices in today's Information Technology (IT) world pose challenges to existing networks in terms of technologies and business models. In this regard, International Telecommunication Union - Radiocommunication Sector (ITUR), which is a specialized agency for facilitating international connectivity in telecommunication networks by allocating global radio spectrum, and developing technical standards to ensure connectivity of network and technology seamlessly, has defined high-level specifications targeting the new generation of mobile communication systems named International Mobile Telecommunications-2020 (IMT-2020) [3].

Unlike the previous generations of mobile communication systems, which had the mission to enhance the network capacity and provide higher speed for the users, International Telecommunication Union (ITU) classifies the demands of IMT-2020 into three broad categories: (i) enhanced Mobile Broadband (eMBB), which intends to meet the users' demand for an increasingly digital lifestyle and focuses on services that have high requirements for bandwidth such as video streaming, online gaming, AR, and VR; (ii) massive Machine Type Communication (mMTC), which is the ability to connect a massive number of sensors to the internet, coined as IoT that is desirable for use case scenarios such as smart cities and smart homes; (iii) Ultra Reliable Low Latency Communications (URLLC), which is required for stringent latency applications such as self-driving, e-health, and industry automation. IMT-2020 requirements include objectives and Key Performance Indicators (KPIs) that any IMT-2020 compliant standard has to meet. For example, IMT-2020 defines that the peak data rate should reach 10 Gbps and even 100 Gbps for special use cases; the number of internet-connected devices should reach 1 million per km^2 and obtain sub-millisecond latency on the air interface [3], [4].

5G has been defined by 3rd Generation Partnership Project (3GPP) as a standard to meet the objectives and KPIs specified in IMT-2020. The major enhancements in 5G is foreseen to happen in the RAN, Transport Network, and Core Network. Several technologies and enablers, such as massive Multiple Input Multiple Output (massive MIMO) and beamforming, Cloud-Radio Access Network (C-RAN), Millimeter Waves (mmWave), Control and User Plane Separation (CUPS), Network Slicing (NS), Service Based Architecture (SBA), MEC, and NFV exist in order to meet the requirements of 5G [3].

MEC [5] and NFV [6] are two key enablers for 5G networks. MEC is a technology that intends to shift the processing, intelligence, and storage resources available in the central cloud towards the network edge, close to the end-users. Employing MEC in the mobile network ecosystem leads to many advantages such as low latency in providing services to the end-users, ability to provide location-aware services, offloading the backhaul, and opening up a new revenue stream for MNOs by leasing their computing resources in the mobile network to third-party business (know as verticals). NFV is another enabler for 5G networks, and its co-existence with MEC can lead to enormous advantages. Employing NFV unleashes the power of virtualization in the context of mobile network and enables the softwarized instance of legacy NFs such as firewalls, IPSs, and IDSs to be executed on Commercial-Off-The-Shelf (COTS) servers. MEC and NFV are complementary concepts. Although the MEC architecture has been designed to work with different deployment options, the most interesting option is the one that allows instantiation of MEC and NFV on the same infrastructure and make it possible to reuse European Telecommunications Standards Institute (ETSI) NFV Management and Orchestration (MANO) components to fulfill management and orchestration tasks [7], [8].

MANO is a key element in the ETSI NFV framework, responsible for the coordination of resources and lifecycle management of network services. Given that resources at the edge are extremely scarce and heterogeneous, developing new approaches that can efficiently utilize network resources and, at the same time, meet user requirements is of great importance. Moreover, network dynamicity is one of the main characteristics of today's networks, which necessitates having approaches that can easily adapt to the network changes and continuously optimize the network [6].

Given the challenges mentioned above, this dissertation aims to study the trade-offs between different SFC placement options and develop optimization-based algorithms to efficiently utilize network resources while optimizing different aspects of the mobile networks. First, we aim to employ the MEC services at the edge in order to collect data about user's behavior and develop ML-based approaches for video content prefetching. Next, we propose new approaches for user association, SFC placement, and VNF migration in the context of a hierarchical 5G network. Finally, we study the trade-off between different VNF scaling approaches and devise an approach that efficiently utilizes network resources while meeting user demands.

1.2 Problem Statement

Caching in mobile networks can be implemented both in the core and RAN. Caching in the core can be implemented through Content Delivery Networks (CDNs), leading to advantages such as ease of management, scalability, and high cache-hit ratio. On the contrary, caching in RAN improves Quality of Experience (QoE) for end-users and alleviates the load on the backhaul link up to 35% [9]. Recently, with the evolution of MEC technology, caching in the RAN can be performed more efficiently. MEC provides storage and computing capacities for the applications in the RAN, close to the end-users. Besides, MEC provides some value-added services for the applications and allows them to make more intelligent caching decisions. An example of such services is the Radio Network Information Service (RNIS) that provides information about the radio context and helps make proper caching decisions. Despite the advantages of caching in the RAN, the storage capacity at the RAN is limited and shared among many applications. Therefore, there is a high demand for devising intelligent methods to cache video contents accurately. **Thus, given the limited computing and storage resources at the edge and considering the finite bandwidth in the backhaul, the problem is to prefetch and cache video segments at the edge efficiently**.

The emergence of MEC as a major enabling technology for 5G networks has made it possible to provide an IT service environment with full cloud computing capabilities within the RAN, in close proximity to mobile end-users. MEC is expected to play a pivotal role in 5G networks by shifting the applications, services, and processing capabilities closer to the end-users and, therefore, offloading the transport network and reducing the round-trip delay experienced by the end-users. For instance, owing to the NFV technology, MEC enables the core network components of the 5G network such as Access and Mobility Management Function (AMF), User Plane Function (UPF), Application Function (AF), and NFs such as firewalls, IDS, IPS, and load balancers to be deployed at the network edge as Virtual Network Functions (VNFs) [10]. Despite the advantages mentioned above, resources at the network edge are very scarce in terms of computing, processing, and storage; therefore, acquiring resources at the edge is very costly. **Thus, another problem is to develop an algorithm that properly associates the User Equipments (UEs) to the gNodeBs (gNBs), embeds the VNFs on the substrate network, and allocates the radio, computing, and transport resources based on the objective defined by the MNOs**.

The rapid change in the mobile data traffic demand calls for efficient approaches to dynamically adjust the mobile network's capacity according to the demand. MNOs have the possibility to increase/decrease the capacity of both the 5G core network and application VNFs upon the need, ensuring optimal resource utilization and lowering the service provisioning cost. This is where the vertical, horizontal, and hybrid VNF scaling strategies come into play. While the vertical VNF scaling implies that the existing VNF is resized upon the need by adding/removing computational, memory, or storage resources, in the case of the horizontal VNF scaling, another instance of the same VNF is spawned/terminated. Although horizontal scaling ensures high scalability and reliability, it suffers from increased resource consumption and state migration challenges. On the other hand, while vertical scaling provides higher utilization of resources, thereby creating resource-optimized VNFs, its lower scalability and inability to change the VNF host significantly affect its practical implementation. Since both scaling strategies have their pros and cons, applying only a vertical or a horizontal scaling strategy cannot perform well in all scenarios. This is why it is important to consider the so-called hybrid VNF scaling strategy, in which it is possible to perform either vertical or horizontal VNF scaling depending on the need. However, it is a non-trivial task to decide which type of scaling to perform for a specific VNF since there is a number of parameters (e.g., the VNF type, its resource requirements) to take into account. After performing VNF scaling, the placement of the VNF is another challenge that requires careful considerations. On the one hand, the interconnections between VNFs composing SFCs must be taken into account in order to make an optimal placement decision. On the other hand, the resource scarcity of the MEC servers at the network edges (e.g., collocated with gNBs) must be considered in order to efficiently utilize the network resources while at the same time satisfying the Quality of Service (QoS) requirement of the requested applications/services. **Thus, the last problem that we tackle in this dissertation is to develop an algorithm that enables the MNOs to dynamically embed SFCs and scale VNFs during the run-time based on the changes in the load.**

All the problems mentioned above are modeled as Virtual Network Embedding (VNE) problems and formulated and solved employing mathematical optimization algorithms. Moreover, heuristic algorithms are proposed to tackle the scalability issues of the mathematical optimization approaches. The following section explains the methodologies adopted to solve the described problems.

1.3 Methodology

Traditionally, networks used to be designed statically with diverse network devices chained in a sequence to provide the desired functionality that the network was designed to deliver. NFV, a major player in the realization of 5G networks, revolutionized the way networks used to be designed. NFV technology decouples NFs from proprietary and vendor-specific hardware; consequently, it enables software instances of NFs called VNFs to be deployed and executed on standard COTS servers [6]. NFV yields numerous advantages, including dynamicity, cost reduction, high availability, service innovation, and reduced power utilization.

One of the most prominent advantages of NFV is the dynamic approach to the construction and management of networks. NFV technology enables MNOs that own the network infrastructure to dynamically share their infrastructure with various Mobile Virtual Network Operators (MVNOs)[1], who can make Virtual Network Requests (VNRs)[2]. MVNOs define their desired services and expectations in the form of VNRs or NF forwarding graphs, which include the network functions and the links that connect these functions [11].

A major challenge in the NFV ecosystem is the VNE problem, which is embedding VNRs in a shared substrate network (physical infrastructure). The VNE problem has been proven to be NP-hard and has been extensively studied by the literature [12]–[14]. The embedding process consists of two steps: node embedding and link embedding. While in the node embedding step, each VNF in the VNR should be mapped into a substrate node, in the link embedding step, each virtual link in the VNR should be mapped to a path in the substrate network.

In this dissertation, we have formulated several VNE problems employing mathematical optimization techniques, which are then solved using the Gurobi mathematical optimization solver [15]. Mathematical optimization models comprise cost minimization/maximization objective(s) and one or more constraints. The cost function is a mathematical formula that determines the productivity of different aspects of the network, such as cost of resources, link utilization, and energy consumption. The model's constraints define the network characteristics and define the limits over either physical and virtual resources. The model needs to fulfill all the constraints to reach a viable solution for the problem. Mathematical optimization solvers use

[1] MVNOs are mobile communication providers that are not owning mobile network infrastructure, but they lease the infrastructure from other MNOs to provide services to their own costumers.

[2] VNRs are simply virtual requests (a.k.a slice in mobile networks) with specific demands in terms of resources requested by, for example, MVNOs.

algorithms such as Branch & Bound (BB) and Branch & Cut (BC) to solve the model. For instance, the BB algorithm starts by removing integrality restrictions to relax the problem. If the resulting relaxed model can satisfy all the integrality restrictions, it can simply reach the solution, and the result will be the optimal solution of the original model. Otherwise, the algorithm further divides the problems into branches (sub-problems) and examines the lower and higher bounds when the solution is fractional. Upon solving one branch, the algorithm stops to examine that branch and compares the results with other branches. In case another branch is found to be superior, the branch will be pruned; otherwise, the best solution will be updated. This process continues until all branches are examined, and the optimal solution to the problem is reached. Unlike the BB algorithm, BC is widely used in the existing mathematical optimization solvers and continuously adds valid inequality constraints (cuts) to the model and seeks to find a valid solution. Usually, cutting branches leads to alleviation of the time required to reach a solution, but sometimes it may have a reverse effect.

While the mathematical optimization techniques always reach the optimal solution to the problem, they become computationally intractable when the problem size increases (e.g., the substrate network components such as gNBs, computing nodes, transport links, and VNR composition of VNFs and the virtual link between them). Aiming to tackle the mathematical models' scalability issue, we propose heuristic algorithms to reach near-optimal solutions in a considerably shorter time scale. As the final step, we confirm the validity of the proposed heuristic algorithms through extensive simulations and comparisons with the optimal solutions derived from the mathematical models.

1.4 Main Contributions

Following the motivations and challenges mentioned in Section 1.1 and Section 1.2, the contributions provided by this dissertation are broadly divided into three main parts:

- Caching video content closer to the users at the edge MEC servers yields several benefits both for the users and the MNOs. Specifically, it curtails the content access delay for the users and improves their QoE [16]. It also alleviates the backhaul transport network load for the MNOs. However, the limited capacity of the edge MEC servers calls for an intelligent decision on what content and where to cache in order to ensure that the QoE of the users

is improved while, at the same time, the network resources (e.g., storage, bandwidth) are used efficiently. In this context, the prediction, anticipatory prefetching, and caching of video segments at the right bitrate during the streaming at the MEC servers play a pivotal role in MEC-enabled Dynamic Adaptive Streaming over HTTP (DASH) video streaming.

Our first contribution, which yielded the following publications [17][3], is to employ machine learning algorithms to predict the number and bitrate of the video segments expected to be requested based on current network bandwidth, playback buffer conditions, and radio network metrics made available by the RNIS at the MEC. We also predict user base station association to detect where to prefetch a requested segment in a changing dynamic network. We then develop a novel formulation to optimize video segment prefetching, transcoding, and resource allocation jointly in a MEC-enabled 5G network. The performance of the proposed approach is shown through extensive simulations performed in various configurations of the network.

- In the context of MEC-enabled 5G networks, not only the edge nodes such as ordinary gNBs can be endowed with computational capabilities, but also the aggregation points of the gNBs (e.g., anchor gNBs) and the core network. The cloud Data Centers (DCs) could still be used for latency-tolerant applications as cheap computational resources. In general, the closer the computing node is to the user, the less its computational capacity is, and the more costly it is to spawn/instantiate VNFs on that node. Given the heterogeneity of computing nodes and the diversity of the QoS requirements (e.g., data rates, latency) of the application, a natural question arises: which gNB should users be associated with and where their required applications should be deployed to ensure that their application requirements are satisfied while the network resources are used in the most efficient manner?

  As our second contribution, which yielded the following publication [18][4], we derive a comprehensive End-to-End (E2E) delay estimation model for users, taking into account the transmission and the propagation time over the air and the transport links along with the VNF processing time. After that, we provide a novel formulation for the joint problem of user association, SFC placement, and resource allocation in the context of MEC-enabled 5G networks. The problem is formulated and solved in a hierarchical mobile network consisting of edge, core, and cloud servers, each with different characteristics and

[3]The extended version of this work is ongoing and soon will be submitted to the IEEE Transactions on Network and Service Management journal, and here we only show partial results.

[4]The extended version of this work is under review in the IEEE Transaction on Network and Service Management journal.

resource provisioning costs. We examine the trade-offs between different user associations and SFC placement options and carry out comprehensive simulations to demonstrate the performance of the proposed approaches.

- As mentioned above, considering the diverse characteristics of the substrate network and a diverse set of applications that have emerged in the context of 5G networks, user association and SFC/VNF placement are very challenging problems that demand careful investigations. Moreover, regarding the dynamicity of mobile networks and the rapid changes in terms of the number of users who use the network and their traffic demand, actively managing and adjusting the network to meet the users' changes in demand plays a fundamental role in today's networking. In this regard, after performing SFC placement, scaling of VNFs is another challenge that requires careful considerations. On the one hand, the interconnections between VNFs composing an SFC must be taken into account in order to make an optimal placement decision. On the other hand, the resource scarcity of the MEC servers at the network edges (e.g., collocated with gNBs) must be considered in order to efficiently utilize the network resources while at the same time satisfying the QoS requirement of the requested applications/services.
  In light of the arguments mentioned above, as our last contribution [19][5], we demonstrate the pros and cons of vertical, horizontal, and hybrid VNF scaling strategies. To this end, we formulate and solve a joint UE association, SFC placement, and VNF scaling problem, having the objective of minimizing the service provisioning cost while satisfying users' QoS requirements. We specifically study the SBA design of the 5G core network and propose a method that embeds and scales different 5G core components, each characterized by different characteristics. Extensive simulations are performed to validate the performance of the proposed method.

1.5 Outline

The structure of this dissertation is summarized as follows. In the current chapter, we first unveiled the motivations behind our work and presented the objectives of this dissertation. The problems, together with the approaches undertaken to solve the problems, were discussed in detail. Finally, the main contributions of this dissertation were highlighted.

[5]The extended version of this work is ongoing and soon will be submitted to the IEEE Transactions on Mobile Computing journal, and here we only show partial results.

Chapter 2 will provide in-depth background on the enabling technologies, including NFV and Software-Defined Networking (SDN), in the context of 5G networks. A particular focus will then be given to the lifecycle management of network services by highlighting in detail the challenges in this research area from different perspectives. Due to the critical role of MEC in our study, a special focus will be given to computing paradigms from cloud to MEC. Eventually, we present a broad explanation of DASH as the main video content delivery standard over the Internet.

Chapter 3 walks through the state-of-the-art studies on the problem of user association, SFC placement, VNF migration, VNF scaling, prefetching, and caching of DASH video content in mobile networks. The key findings of the state-of-the-art works in each of these challenging problems will be presented.

In chapter 4, we investigate the problem of DASH video content prefetching in mobile networks. Firstly, we define the motivations behind video content prefetching and caching in mobile networks and the need for anticipatory prefetching of video segments at the right bitrate during streaming at the MEC servers. In this regard, we employ ML algorithms to predict the number and bitrate of the video segments expected to be requested based on current network bandwidth, playback buffer conditions, and radio network metrics made available by the RNIS service at the MEC. We also predict user base station association to detect where to prefetch predicted segments when the user associations change. We then formulate an ILP-based problem for jointly optimizing video segment prefetching, transcoding, and resource allocation, with the objectives of maximizing cache-hit and byte-hit ratios.

In chapter 5, we study the joint problems of user association, SFC placement, and resource allocation in MEC-enabled 5G networks. Specifically, we first motivate the need for having a system that optimizes resource utilization for MNOs, and at the same time, meets user demands given the heterogeneity and varying cost of resources in the network. We then present MILP techniques to provide novel formulations of the problem. The study proposes three minimization objectives, including service provisioning cost, the impact of VNF migration on UE's experienced QoE, and transport network utilization. We also develop a scalable heuristic algorithm that reaches a near-optimal solution to minimize the impact of VNF migration on QoE of UEs in a much shorter time scale compared to our proposed MILP-based algorithm. We perform comprehensive simulations, drawing a comparison between the proposed algorithms by considering different types of service requests with diverse data rates and E2E latency requirements.

In chapter 6, we investigate the joint problems of user association, SFC placement, and scaling of 5G core components in MEC-enabled 5G networks. Firstly, we motivate the need for having efficient approaches to adjust the mobile network's capacity according to the demands. Our approach enables MNOs to increase/decrease the capacity of both the 5G core network and application VNFs upon the need, ensuring optimal resource utilization and lowering the service provisioning cost. We thus propose a method that demonstrates the pros and cons of vertical, horizontal, and hybrid VNF scaling strategies. To this end, we formulate and solve a joint UE association, SFC placement, and VNF scaling problem by leveraging ILP techniques, with the objective of minimizing the service provisioning cost while satisfying users' QoS requirements. A scalable heuristic method will be proposed to address the scalability issue of the ILP formulation. The performance of the presented algorithms will be compared by extensive simulations carried out considering different types of service/requests with diverse requirements.

Finally, chapter 7 summarizes the main contributions of this dissertation, followed by the key findings of the work. Moreover, we propose several promising research directions for future works.

Chapter 2

Background

2.1 SDN and NFV in 5G Network

Virtualization is a technique widely used in IT and specifically in the cloud domain to create isolated logical components on top of a standard and shared physical infrastructure. Virtualization makes system design considerably more flexible, agile, and efficient compared to the traditional fixed and hardware-dependent approaches. Recently, virtualization has found great applicability in mobile networks [20]. With the emergence of 5G networks, virtualization became a promising technology to fulfill the defined 5G objectives. SDN and NFV are two major enablers for 5G networks that are based on network abstraction. While SDN aims to separate control and data planes and deliver centralized management to the network, NFV's mission is to decouple NFs from hardware and make them able to run on standard and vendor-agnostic hardware. Although SDN and NFV can exist without each other, their co-existence results in more significant benefits. Nguyen et al. [21] provide a detailed survey on the applications of SDN and NFV in mobile networks.

SDN technology has gained tremendous attention during the last decade. The separation of control and data planes is the principal strategy of SDN towards reaching programmability and centralized management in the network [22]. A general view of SDN architecture comprises three planes: data plane, control plane, and management plane. SDN has completely transformed the legacy vertical network design, and instead, brought forward a horizontal approach in which development and innovation are more likely to happen. This transformation breaks the guidelines set in the classical networking approach, in which both control and data planes are located in the physical devices, developing protocols is vendor-dependent, and network

configuration is cumbersome. Overall, an SDN-based architecture is differentiated from the traditional network architectures by having four characteristics of decoupled control and data planes, centralized network decisions, flow-based data forwarding, and abstracted management policies [23].

The data plane in SDN includes network devices (routers, firewalls, load balancers, etc.) and is responsible for forwarding data flows in the network. These devices do not perform any forwarding decision; instead, they follow the instructions received from the control plane. The control plane is the location where decisions about routing, switching, and policies takes place. It also contains protocols that determine the behavior of the network. The control plane (also known as network operating system) is responsible for converting the high-level decisions received from the management plane into the low-level rules that can be executed by the underlying devices [23].

The plane separation requires well-defined Application Programming Interfaces (APIs) to establish communication between the planes [23], [24]. Northbound APIs are programming interfaces that allow the applications and orchestrator in the management layer to define a high-level policy for the network. Then, the control plane converts the policies into more specific rules that can be implemented into the commodity hardware through southbound APIs such as OpenFlow [25], [26], OVSDB [27], and ForCES [28]. Detailed studies on SDN architecture and protocols can be found in [22]–[24], [29], [30].

The NFV technology plays a key role in the realization of the 5G networks [6], [31], [32], as it decouples the legacy Network Functions (NFs) such as routers, firewalls, IDS, IPS, etc., from purpose-built hardware and deploys them as platform-independent VNFs. Apart from the legacy NFs, 5G core components such as the UPF, AMF, and Session Management Function (SMF) in the 5G core SBA design [33] are another example of NFs that can be deployed as VNFs, providing unprecedented management flexibility while curtailing both Capital Expenditure (CapEx) and Operational Expenditure (OpEx). Furthermore, the separation of functionality from hardware promotes evolution in both software and hardware planes independently. Additionally, NFV provides the opportunity to represent NFs and applications as a single VNF or multiple VNFs interconnected in a particular order forming SFCs.

2.2 VNF Lifecycle Management and Service Function Chaining

Ensuring a highly available framework that can provide E2E service provisioning requires an efficient system to manage, orchestrate, monitor, and control system components. In this regard, a significant effort from standardization bodies has been devoted to developing an NFV management and orchestration framework. ETSI has realized the high potentials of NFV and allocated an Industry Specification Group (ISG) to develop a prevalent NFV framework called NFV MANO [6].

In the ETSI NFV MANO framework, VNFs are software implementations of legacy NFs that run on top of the NFV Infrastructure (NFVI), which by itself comprises a virtualization layer that provides virtual resources from a pool of physical resources to make VNFs run on a logically isolated shared infrastructure. NFV MANO is the entity responsible for making sure that services requested by the users are up and running. It includes three modules: Virtualized Infrastructure Manager (VIM), VNF manager (VNF), and NFV Orchestrator (NFVO). The VIM constitutes the functionalities for controlling and managing the interaction between VNFs with the physical and virtualized resources. The VNF manager is in charge of the lifecycle management of VNFs. Finally, the NFVO is responsible for orchestration and management of physical and software components and realizing services on NFVI. Lifecycle management includes instantiation, updating, migration, scaling, and termination of VNF instances to ensure service continuity during the runtime of the VNFs. A detailed description of the framework and the reference points can be found in [6].

Placement. One of the main intentions of the NFV MANO platform is to have better resource utilization by allocating multiple VNFs to the same physical resource or aggregating multiple physical resources to serve a function with higher demands. Allocating VNFs to physical resources is a challenging task that should be performed wisely. Translating high-level SFC placement goals into low-level instruction happens in the NFVO component in the MANO framework. The NFVO component should contain an algorithm that is capable of mapping SFCs to physical hardware dynamically [34]. Finding an optimal placement of SFCs on the substrate network is one of the challenges that this dissertation will address.

Migration. Regarding the dynamicity of current networks, migration of VNFs is one of the main challenges that need to be studied. There are many scenarios in which the migration of VNFs is required or may result in better performance. For

instance, a user that receives services from one base station easily moves to another base station, and the management system should support its migration without worrying about service disruptions. Moreover, it is possible that adding a network function to the network triggers some topology changes in the network. Besides, in some cases moving VNFs to different hosts can boost the overall network performance [35].

Scaling. The rapid changes in the mobile data traffic volume call for efficient approaches to dynamically adjust the mobile network's capacity according to the demand. The scalability feature of NFV technology enables MNOs to meet the user demands in real-time and makes them able to handle unexpectedly high peak loads when needed without over-provisioning of resources. Therefore, the resources to meet the growing demand of users can be provisioned immediately without the lead times inherent in planning deployments of network hardware appliances or the costs associated with specialized installation procedures. MNOs will increase/decrease the network capacity of the 5G core and application VNFs upon the need, ensuring optimal resource utilization and decreasing the service provisioning cost. This is where the vertical, horizontal, and hybrid VNF scaling strategies come into play. While vertical VNF scaling implies that the existing VNF is resized upon the need by adding/removing computational, memory, or storage resources, in the case of the horizontal VNF scaling, another instance of the same VNF is spawned/terminated.

2.3 Multi-access Edge Computing

DCs are situated at the core of concentration in modern-day software development paradigms. DCs are the primary building block of the IT infrastructure for numerous small to large-size organizations and businesses. The traditional approach of utilizing DCs in organizations was to keep silos of physical storage and computing devices in the organization premises, aiming to preserve data in its premises. However, the necessity of updating the hardware to meet the users' demand is a limiting factor against DC growth. Furthermore, storing one organization's data in one place brings up many security and vulnerability issues; hence, the DC approach has been progressively replaced by cloud computing.

Cloud computing is a paradigm referring to the on-demand delivery of computing, storage, networking, and software resources in a pay-as-you-go manner. Thanks to virtualization technology, a pool of resources is available to the users and can be accessed and released dynamically with minimal required management efforts [36]. Many businesses have adopted the cloud computing paradigm due to

the enormous advantages it brings to their organizations. The ease of management, accessibility, the flexibility of growth, and cost per usage are some advantages of cloud computing, to name a few.

Many newly emerged applications and services that adopt Artificial Intelligence (AI) and big data analytics are tailored to a cloud-based approach for their data processing. Despite the extensive advantages of cloud computing, some challenges call for novel technologies to evolve and address the gaps. One of the main issues with cloud computing adoption for delay-sensitive applications is the long distance between the data source (cloud) and data consumer (e.g., user, devices, sensors). Moreover, the transmission of massive data to/from the cloud causes extra backhaul utilization [37]. Location unawareness is another issue with the centralized-cloud approach. By performing all the computing in the cloud, tracking the usage patterns of users belonging to distinct geographical areas is difficult.

MEC [7] is one of such technologies that is expected to play a principal role in 5G networks by shifting the applications, services, and processing capabilities closer to the end-users and, therefore, offloading the transport network and reducing the round-trip delay experienced by the end-users. MEC is expected to be widely adopted in the 5G networks to satisfy the ultra-low latency requirement of certain applications and services while at the same time alleviating the transport network load [7].

MEC employs virtualization in order to run MEC applications as software-only entities at the mobile network edge on top of the virtualized infrastructure. As stated earlier, NFV provides a virtualization platform, where NFs can run on top of it as software instances and being managed and orchestrated. MEC and NFV are two complementary concepts, and MEC architecture is designed in a way that can support different deployment options. A deployment option of MEC is proposed by ETSI that allows instantiation of MEC application and VNFs on the same virtualization infrastructure. The deployment proposes using ETSI NFV MANO to perform the management and orchestration of MEC applications [7].

The infrastructure that hosts both MEC applications and VNFs is very similar; therefore, ETSI proposed a MEC architecture that can host both VNFs and MEC applications on the same infrastructure [7]. Apart from that, there are some other essential functionalities that MEC is anticipated to fulfill. Mobility support is an inevitable functionality for mobile networks, which is expected to be considered in the design of MEC. In order to support mobility functionality, a MEC system needs to support continuity of the service, migration of applications/VNFs, and transfer

and exchange of application and user-specific states. ETSI has considered different deployment scenarios for MEC, given the performance, cost, scalability, and operator preferred deployments. For instance, MEC can be deployed in the RAN, at an aggregation point, or at the edge of the core network.

Both MEC applications and VNFs have various resource requirements regarding computing, storage, and network resources. Moreover, applications can have different QoS requirements, for example, strict latency or data rate requirements. Plus, for many applications, the condition may vary over time and require the MEC system to change the location of the application due to mobility of the UEs, lack of resources, and energy efficiency. Given these reasons, the MEC system should be able to deploy VNFs and MEC applications at the most suitable node and at the right moment. Moreover, the MEC system should be able to react to variations that happen in the network status and, based on the changes, place, migrate, and scale application VNFs.

2.4 Prefetching and Caching

With the emergence of 5G high-speed networks, the expectation for high-quality 4K/8K video streaming has also increased. The emergence of MEC technology enables MNOs to provide network services at the edge [9]. The effective handling of video traffic at the mobile edge by MNOs will become increasingly important to satisfy customers with QoS as they stream higher volumes at higher qualities.

Currently, HTTP Adaptive Streaming (HAS) is the dominant video delivery technique and has been adopted by leading video content providers such as YouTube and Netflix [38]. Videos in the HAS technique are split into equally sized segments available in multiple bitrates (qualities). DASH [39] is a standardized HAS technique, which emerged as a collaboration between 3GPP in TS 26.234 Release 9 [40] and Moving Picture Experts Group (MPEG) [41]. The metadata about the segments and bitrates in DASH is available in the Media Presentation Description (MPD) at the video server, accessible to the clients upon issuing a video request. The Adaptive Bitrate (ABR) algorithm in the client continuously measures the bandwidth and buffer status. Based on the measurements and the MPD file, the client makes the next segment's request in a way that the user can achieve the highest satisfaction. The user's satisfaction is achievable through reaching minimum stalling at the playback, minimum bitrate oscillation, and maximum usage of the bandwidth [42].

Caching video content closer to the users at the edge MEC servers yields a number of benefits both for the users as well as for the MNOs. Specifically, it curtails the content access delay for the users and improves their QoE [16]. It also alleviates the backhaul transport network load for the MNOs. The limited capacity of the edge MEC servers, however, calls for intelligent decisions on what content and where to cache, so to make sure that the users' QoE is improved while the network resources (e.g., storage, bandwidth) are used in an efficient manner. In this context, the prediction, anticipatory pre-fetching, and caching of video segments of the right quality during streaming at the MEC servers play a key role in MEC-enabled DASH video streaming.

Chapter 3

State of the Art

3.1 DASH Video Prefetching and Caching

A considerable portion of the Internet traffic originates from the excessive download of a small set of popular multimedia content requested by a vast number of users. In-network caching helps avoid retransmission of duplicated popular contents, leading to savings in transport bandwidth, backhaul bandwidth, energy usage, and improving QoE for the end-users. In this regard, a sizeable body of literature has studied the problem of DASH video caching in various network deployment scenarios.

Liang et al., in [16] proposes an online algorithm able to prefetch and cache video segments in real-time, aiming to maximize the byte-hit ratio under the limited bandwidth condition between the proxy and video server. Authors in [43], aiming to maximize the byte-hit by prefetching video segments, propose a framework that predicts the next segment bitrate requested by the user, taking the cache server bandwidth and the client's adaption scheme information into account. The study in [44] formulates the problem of video bitrate selection as a MILP model to maximize user utility. Moreover, the authors study the effect of multi-path on the transmission capacity in a multi-autonomous system environment. Finally, a distributed algorithm capable of reaching a near-optimal solution on a shorter time scale is proposed to tackle the scalability issues of the MILP-based algorithm.

Authors in [45] consider a mobile network, enabled with MEC servers that can expose the network information service to be used for service improvement. An adaption algorithm runs on the MEC servers, responsible for alleviating network congestion and improving the user's QoE. The study in [46] considers a heterogeneous network in which each client can switch between different wireless networks.

A MEC application is introduced to estimate the bandwidth and update the client about the network condition. Therefore, it is considered as a proxy-based approach that helps the client to make proper decisions at the moment.

Authors in [47] propose a MEC-enabled framework capable of utilizing RNIS to cache and update the cache for DASH video service. They propose request popularity and expected popularity as two metrics to improve video quality and reduce buffer time. The work in [48] proposes a MEC-based DASH video caching strategy. The algorithm stores each segment's highest bitrate on the edge servers and employs the processing power accessible at the MEC servers to transcode the video segment on demand. A cache prefetching scheme is proposed in [49], which is able to prefetch video segments using an adaptation algorithm that considers the throughput measurements from the client and the predicted throughput at the cache. A learning-based caching and the prefetching method is proposed in [50] to improve users' QoE for adaptive video streaming. The algorithm caches the most popular video segments at the edge to tackle the problems of network jitter.

Performing prediction in RANs is especially challenging due to continuous changes in the physical channel conditions and the availability of different RANs [51]. Employing ML to predict specific metrics (e.g., channel throughput) for RAN has gained importance in the past years [51]–[54]. Within the context of DASH, previous work employing ML focuses mainly on bandwidth estimation at the client, which constitutes an input to most ABR algorithms. To this extent, Raca et al. [55] demonstrate that integrating throughput prediction in the client can increase QoE regardless of the employed ABR algorithm. The same authors [56] further explore this idea by employing the random forest algorithm at the client to predict the expected average throughput over a time horizon. The model employs percentiles of historical client-side radio channel metrics (e.g., RSRP, and SNR) and historical application throughput as input to the model. Another work by [56] evaluates the influence of different window sizes for both metrics aggregation and prediction horizons on a dataset with different mobility patterns and test their model using different ABR algorithms. Overall, their model achieves a QoE improvement by up to 30%. Moreover, Mao et al. in [57] develop a reinforcement learning method for directly obtaining the bitrate for the next video chunk. The model employs an Actor-Critic neural network model at the client, whose input includes historical throughput information, buffer state, and next chunk sizes. Lian et al. [16] demonstrate and motivate the benefits of predictive pre-fetching. They consider streaming over a wired network wherein the segment bitrate switching rate is low, justifying their assumption that the next bitrate requested can be assumed to be the same as the previous bitrate requested.

3.2 User Association

The problem of user association in 5G networks is one of the research areas investigated in this dissertation. An optimal user association mechanism results in an efficient PRB utilization at gNBs, while ensuring the expected QoE for the users [58]. User association in mobile networks and particularly in 5G network has been the center of focus for many studies [59]–[67].

Liu et al. in [59] formulate the problem of user association in HetNets as a Nash bargaining problem. The objective is to maximize data rate utility while guaranteeing users' data rate demand and balance load among the base stations. Lei et al. in [60] design a delay-aware user association strategy for 5G HetNets intending to minimize the overall power consumption in the network while applying strict delay constraints. Amine et al. in [61] formulate the problem of user association in 5G ultra-dense multi-RAT HetNets as a multi-objective optimization problem, which is solved leveraging the weighted sum technique. The work by Cacciapuoti et al. [62] presents a constrained optimization method for mobility-aware user association in mmWave networks. The method is capable of tracking the frequent variations in the network topology and channel condition. Similarly, the work by Amine et al. [63] addresses the UE association problem in 5G HetNets to meet the user's QoE requirements using a one-to-many matching game based on matching theory.

The work presented by Goyal et al. [64] introduces an optimal user association method in 5G mmWave networks, which can recalculate the cost of possible handovers and also the erratic nature of mmWave channels. The work by Harutyunyan et al. [65] studies the user association problem in a cache-enabled mobile network, capturing the trade-off between the radio access network and the transport network utilization in 5G networks. A joint user association and user scheduling solution has been presented by Ge et al. [66], where the authors aim to minimize the users' achievable throughput. Liakopoulos et al. [67] employ a data-driven technique to predict future traffic patterns and then associate users with base stations based on pre-calculated association maps of the given time.

3.3 SFC Placement

The service function chain placement is yet another problem studied in this dissertation. There is a sizable body of works studying the SFC problem [68]–[80]. Moreover, a vast number of surveys fully explore this problem from different perspectives,

such as type of required placement (i.e., dynamic or static), objectives, and metrics of the VNFs [81]–[83].

The study by Alleg [68] addresses the problem of SFC placement to efficiently utilize the network resources while respecting the E2E latency requirement of the UEs. Zhang et al. [69] propose a VNF placement method, which takes advantage of the edge, core, and cloud servers in service-customized 5G networks. An interference-aware method is proposed to tackle the negative effect of the VNF consolidation (i.e., VNF interference) with the goal of maximizing the overall throughput of the accepted requests. Yang et al. [70] provide two models to calculate, respectively, the transmission delay of flows traversing a chain of VNFs and the availability of SFC for VNF resiliency. Furthermore, they propose an Integer Non-Linear Programming (INLP) model and a heuristic algorithm to jointly solve the problems of delay-sensitive VNF placement and VNF resiliency. Similarly, the approach in [71] solves an SFC-based resource allocation problem using ILP by jointly tackling the VNF placement and routing problems to reduce energy consumption. The same problem is investigated by Wang et al. in [72], where MILP techniques are used through a three-phase study, namely VNF chain composition, VNF forwarding graph embedding, and VNF scheduling. Agrawal et al. [73] jointly solve the problems of VNF placement and CPU allocation in 5G networks. The authors consider latency as the primary KPI and try to minimize the ratio between the actual and maximum allowed latency.

The work presented by Zhang et al. [74] utilizes the theory of open Jackson network to evaluate data traffic in data centers and proposes two heuristic algorithms to jointly optimize the SFC placement and request scheduling while minimizing the latency and resource utilization in the network. Similarly, the study in [75] proposes a MILP model for VNF placement in hierarchical 5G networks, where VNFs can be deployed at the edge, core, and cloud nodes. The main goal is to minimize the overall latency, which is composed of queuing, processing, transmission, propagation, and optical-electronic-optical conversion delay. The parallel VNF deployment approach is adopted in [76] to achieve latency reduction in service delivery. The bottleneck issue caused by the imbalanced deployment of parallel VNFs is mitigated by mapping multiple instances of the VNFs. Moens et al. [77] introduce an ILP model to map VNFs on the servers in order to minimize the number of utilized servers. The work, however, does not consider the underlying network characteristics but only services and VM requests. The work by Bari et al. [78] investigates a VNF Orchestration Problem (VNF-OP) and proposes an ILP and a heuristic solution to determine the number of required VNFs and their locations without violating Service

Level Agreements (SLAs). The main objective of the work is to minimize OpEx and resource fragmentation. The authors of [79] jointly study the problem of VNF placement and routing, having the objective of maximizing network throughput. Finally, the proposed work in [80] jointly tackles VNF placement and resource allocation problems as a Mixed-Integer Program (MIP) based on an SDN/NFV-enabled MEC infrastructure. However, the fitness function does not consider E2E service latency requirements. The work in [84] formulates the problem of VNF placement at the network edge to minimize the network latency from the UEs to their respective VNF hosted on edge servers. A method is presented to dynamically re-schedule VNFs so as to attain optimal allocation and avoid SLA violations. The study by [85] presents an ILP model to jointly solve the problems of user association, SFC placement, and resource allocation, in which UEs are assumed to have different E2E latency and data rate requirements.

3.4 VNF Lifecycle Management

3.4.1 VNF Migration

VNF migration is yet another interesting problem that is covered in this dissertation. There is a long line of research attempting to address the VNF migration problem [35], [86]–[89].

The study by Xia et al. [35] defines the VNF migration cost as the overall traffic served by the VNF, which is minimized by an ILP model. Furthermore, trying to tackle the scalability issue of the ILP model, a heuristic model is proposed to minimize the migration cost and satisfy the computing and transport resource utilization constraints. Another study in [86] models the problem of VNF migration for latency stringent applications in a highly dynamic environment. The work proposes a heuristic algorithm that triggers the VNF migration based on the applications' latency requirement violation. Carpio et al. [87] introduce a linear programming model to combat the problems of QoS degradation caused by service interruptions and improper load distribution among servers. They study the trade-off between VNF replication and migration of already deployed VNFs to balance the load on servers and reduce migrations. The work presented by Hawilo et al. [88] proposes a MILP model to smartly decide whether to migrate or instantiate the VNF of the same service in case of failure or resource scaling, with the objective of minimizing service downtime and service latency. The work presented in [89] considers flexible placement and

migration of VNFs in a MEC-enabled 5G architecture. The authors consider both the computational and network needs of the UEs and present a proof of concept where the NFV orchestrator handles network resources and services in real-time.

3.4.2 VNF Scaling

Significant research effort has been invested in studying the problem of VNF scaling [90]–[98]. In [90], Sedaghat et al. study the trade-off between cost and performance for vertical and horizontal scaling of virtual machines. The study in [91] evaluates the performance of horizontal, vertical, and hybrid scaling in the cloud environment, concluding that while horizontal scaling has the lowest overhead, the hybrid method offers the highest flexibility. Furthermore, Wang et al. in [92] conclude that the VM scale-out operation is preferable for cloud environments under low throughput demand and budget constraints, while the VM scale-up operation is more appealing for high throughput demand.

Buyakar et al. in [93] propose a vertical auto-scaling algorithm for data plane VNFs in the 4G core network to avoid under-utilization of network slices. On the contrary, a control theory-based VNF horizontal scaling method is put forward in [94], which also studies load balancing among AMF instances. A fixed threshold-based vertical scaling approach is introduced by Moghaddassian et al. in [95]. The resource threshold is updated based on real-time monitoring of the data utilization, while the scaling-down/up decisions are made during an observation period. A queuing theory-based and a mathematical model are proposed in [96], which formulates an optimization problem that aims to minimize the VNF's processing time and link transmission delay. Both vertical and horizontal VNF scaling strategies are separately considered. Tang et al. [97] study the horizontal VNF scaling problem proposing an approach for forecasting the load on the VNFs in order to scale them on time. Subramanya et al. [98] present an ML-based method for predicting the number of VNFs needed and then employ ILP techniques to place and scale VNF instances.

Chapter 4

MEC-Assisted Video Prefetching

In this chapter, we propose an approach for MNOs to efficiently use the infrastructure with minimal resource over-utilization and maximum user satisfaction for the DASH video streaming application. Specifically, we employ machine learning algorithms to predict the number and bitrate of the video segments expected to be requested based on current network bandwidth, playback buffer conditions, and radio network metrics made available by the RNIS at the MEC. We also predict user base station association to know where to prefetch a requested segment in a changing dynamic network. We then formulate an ILP-based model for jointly optimizing video segment prefetching, transcoding, and resource allocation having objectives of maximizing the cache-hit and the byte-hit ratios. A heuristic algorithm will be proposed to tackle the scalability issue of the ILP-based approach, achieving a near-optimal solution in a considerably shorter time scale while maximizing the cache-hit ratio.

4.1 Overview

Video content is the dominant traffic in terms of volume on the current Internet. Cisco forecasts that video will constitute roughly 79% of the total Internet traffic by the end of 2022 [2]. With the ongoing deployment of 5G high-speed networks, the expectation for high-quality 4K/8K video streaming over mobile networks has also increased. The emerging MEC technology enables MNOs to provide network services at the mobile edge [9]. The effective handling of video traffic at the edge by MNOs will become increasingly important to satisfy customers with the guaranteed video QoE, as they stream higher volumes at higher qualities. A video streaming

use-case, such as live streaming events at a sports festival, is an example of a challenging scenario wherein a large number of users stream videos in a small area, streaming instant replays, and live streams of other games at sports festivals. In the future, this could even include live-VR with stricter QoE requirements.

Currently, DASH [39] is the dominant video delivery standard that has been adopted by most content providers, e.g., YouTube and Netflix [38]. It dictates that each video is split into equally-sized segments available at multiple video qualities, or *bitrates*. High variability of the bandwidth available to a user in dynamic mobile networks requires an adjustment of the video bitrate based on the network and playback buffer's current state. This is done by the ABR algorithm, which, using monitoring information, adjusts the bitrate of the next segment request to maintain the highest possible QoE for the users [42].

Caching video content closer to the users at the edge MEC servers yields benefits both for the users and the MNOs. It reduces the content access delay for the users, improving their QoE [16] while also alleviating the backhaul transport network load for the MNOs. However, the limited capacity of the edge MEC servers calls for intelligent decisions on what content to cache and where to cache it to improve QoE while also efficiently using the network resources. In this context, prediction, anticipatory prefetching, and caching of video segments at right bitrate during streaming at the MEC servers play a pivotal role in MEC-enabled DASH video streaming.

In this chapter, we employ machine learning algorithms to predict the number and bitrate of the video segments expected to be requested based on current network bandwidth, playback buffer conditions, and radio network metrics made available by the RNIS at the MEC. We also predict user base station association to know where to prefetch a predicted segment when the user associations are changing. We then formulate an ILP based problem for jointly optimizing video segment prefetching, transcoding, and resource allocation, with the objectives of maximizing cache-hit and byte-hit ratios. Moreover, a heuristic algorithm is proposed to tackle the scalability issue of the ILP-based solution, achieving a near-optimal solution in a considerably shorter time scale while maximizing the cache-hit ratio.

4.2 Problem Statement

Figure 4.1 depicts the envisioned mobile network, which is composed of a 5GC, and gNBs, with MEC servers co-located with gNBs. The MEC servers are characterized by processing, memory, and storage resources. While these resources are limited for the MEC servers at the network edge, they are abundant at the 5GC. Therefore, the resource usage at the 5GC is much cheaper than that of the network edges, though the former also imposes transport network usage costs.

We assume that a set of requests will be issued from UEs for a set of video segments, possibly in different bitrates at any given time. The ML model proposed in Section 4.4.1 is responsible for predicting the requests, the bitrate, and the gNB association of each UE. After obtaining the prediction outcome from the ML model, the ILP model decides whether to pre-fetch the requested video segment(s) in specific bitrates to the network edge or, if available, to transcode higher bitrate segments available at the network edge to make sure that the bitrate requirements of UEs are satisfied while the network resources are used efficiently.

Depending on the segment duration, segment bitrate, and availability of the substrate network resources, there might be multiple pre-fetching options, each in favor of optimizing certain aspects of the network. The problem of joint video segment pre-fetching, transcoding, and resource allocation can be formally stated as follows.

Given: a 5G network composed of MEC servers collocated with gNBs and the 5GC, which are interconnected via transport network links. Additionally, a set of UEs that are connected to gNBs, making video segment requests with specific bitrates.

Find: joint video segment pre-fetching, transcoding, and resource allocation.

Objective: maximize (i) the cache-hit ratio and (ii) the byte-hit ratio. We define the cache-hit ratio as the number of requests served from the edge (whether directly from the same gNB, from neighbor gNBs, or using transcoding) to the number of requests issued to the network. Similarly, the byte-hit ratio is defined as the number of bytes served from the edge to the number of bytes requested by the UEs.

4.3 System Architecture

Fig. 4.2 illustrates the system architecture. It consists of three main modules: 5G Core (5GC), MEC Server, and DASH Client. The 5GC server contains the Video

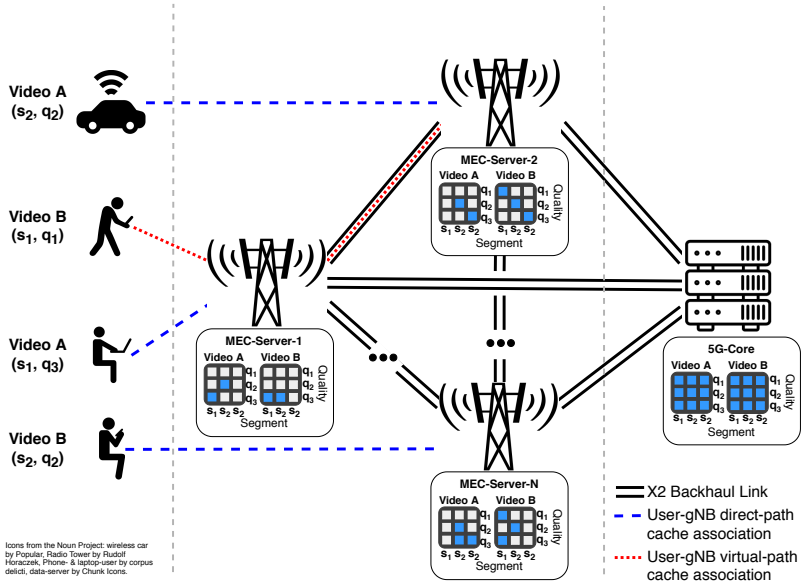


FIGURE 4.1: Sample mobile network and service request models.

Server and the Decision Maker sub-modules. The Video Server provides all the segments of available videos and a MPD file containing metadata about each video's available representations. The MPD file is available to the clients upon requesting a video to ensure that they only ask for the available bitrates on the video server.

The Video Server provides the videos to the Prefetch Requester or directly to the client through the Request Handler. A client can be directly served from the Video Server when the decision given to the Request Handler mandates not to prefetch the requested segment or when one of the prediction models is wrong. Decision Maker implements the logic of the ILP and Heuristic algorithms. The algorithms run periodically on the 5GC upon receiving the prediction output during each prediction window.

The MEC Server encompasses several sub-modules. The RNIS sub-module collects radio metrics related to each client and then feeds them to the ML Predictor sub-module. The ML Predictor uses the data received from the RNIS and Request Handler to predict the next segment(s) bitrate(s) for the clients. The Request Handler takes the prediction's output to query the Cache Manager to check if the predicted segment bitrate can be found in the cache. If the local cache cannot fulfill the request, the Decision Maker will be asked to decide on the request. Upon making a decision, it notifies the Request Handler about how to handle the request. There are four possible options for the Request Handler if the content cannot be found in the local cache: (i) prefetch the segment with the requested bitrate using the Prefetcher sub-module, (ii) prefetch a higher bitrate of the segment and use it for serving even the users that

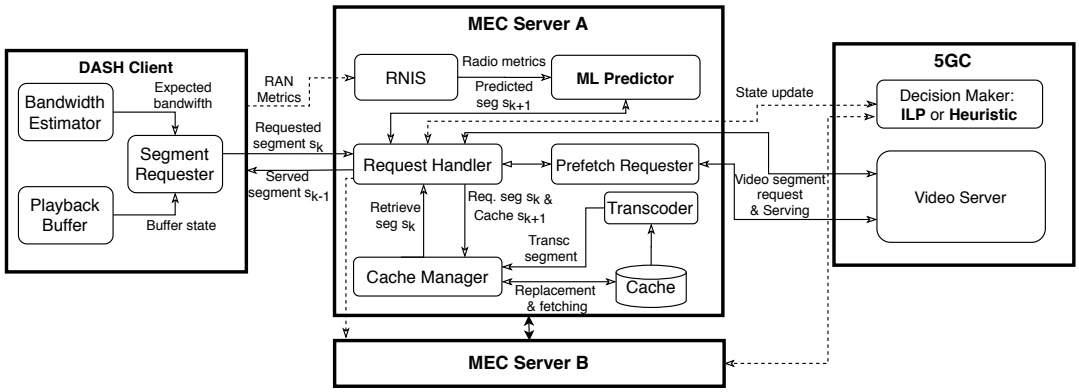


FIGURE 4.2: Proposed system architecture.

ask for the lower bitrate of the same segment using the Transcoder sub-module, (iii) redirect the request to the Video Server at the 5GC, or (iv) redirect the request to one of the neighbors MEC Servers that can have the same bitrate or even a higher bitrate that can be transcoded. It is essential to mention that this process is performed after handling the request for segment (s_k), before the next segment (s_{k+1}) is requested.

Finally, the last module, DASH Client, comprises three sub-modules, Bandwidth Estimator, Playback Buffer, and Segment Requester. The Segment Requester collects the information from the other two sub-modules and decides the next segment's bitrate to be requested. The request will then be issued to the Request Handler in the MEC Server.

4.4 Proposed Methods

4.4.1 Prediction Model

The prediction algorithm runs periodically over prediction windows to predict the requests expected in the next window. The ILP decides which of these requests to pre-fetch at the beginning of each window (decision instant). The goal of the prediction model is to provide the ILP with information at each decision instant, about what segment requests are to be expected from the clients in the next window. Assuming sequential segment requests by the client, an effective caching strategy requires for each client the prediction of both the qualities of the requested segments over the *prediction window*, and the MEC server node in the network at which to place the content. The prediction procedure runs synchronously at each decision instant before the ILP, whereby one prediction is obtained for each client associated with the respective MEC server. The prediction window size is denoted as Δt. At

any given decision instant k, RAN and DASH client-related metrics are collected over the metrics aggregation window, i.e., over the time interval $[t_k - \theta\Delta t, t_k)$, $\theta \in \mathscr{R}^+$. These metrics are fed to the predictor model (see Fig. 4.3). Since the ILP runs synchronously, the prediction output $P_{n,k}$ for each client n at the decision instant k must include the number of segments expected to be requested by each client over the prediction window, the expected bitrate of these segments, and the expected gNB association to place the content. The modular design of the system (see Fig. 4.2) allows instantiating the predictor with different algorithms. In this work, we devise an ML system composed of three individual predictors, as described below.

Number of Segments: The *NSEG-Predictor* returns the expected number of segments requested by the client n ($N_{n,k}$ in Fig. 4.3-III). Configurations at the DASH client and the current state of the playback buffer determine the client's number of requests (including no requests). Hence, $N_{n,k}$ is an integer number greater than or equal to 0 ($N_{n,k} \in \mathscr{N}_0$).

Bitrate Mode: The *MODE-Predictor* returns the most frequent bitrate of the requested segments for client n ($Q_{n,k}$ in Fig. 4.3-III). One bitrate $Q_{n,k}$ is predicted and assigned to the $N_{n,k}$ segments expected to be requested. In the case of multiple modes, the highest bitrate is selected due to the transcoding option at the MEC. MODE-Predictor performs predictions only on samples with $N_{n,k} \geq 1$.

gNB Association: The *GNB-Predictor* returns the expected gNB association for client n ($B_{n,k}$ in Fig. 4.3-III). gNB association is relevant when considering client mobility since it determines where to place the pre-fetched content. Wrongly allocating MEC capacity for content will negatively affect the cache-hit rate.

Each predictor handles a multi-class classification problem. The number of possible classes for NSEG- and MODE-predictor depends on the choice of Δt. For the GNB-Predictor, it depends on the number of MEC edge servers in the network.

Figure 4.3 summarizes the metrics used for the three prediction tasks. The input metrics used to train the NSEG- and MODE-predictors either directly or indirectly influence the bitrate chosen by ABR algorithms (a,b,c,d,e from Fig. 4.3-I). The input metrics to the GNB-Predictor influence the clients gNB association (b,g,f from Fig. 4.3-I). Since only averages over windows do not capture the distribution of some metrics, we also use the 25th, 50th, 75th, and 90th quantiles for metrics a,b and d.

These metrics can be obtained at the MEC server through the RNIS and a monitoring component that logs the current state of each UEs DASH client based

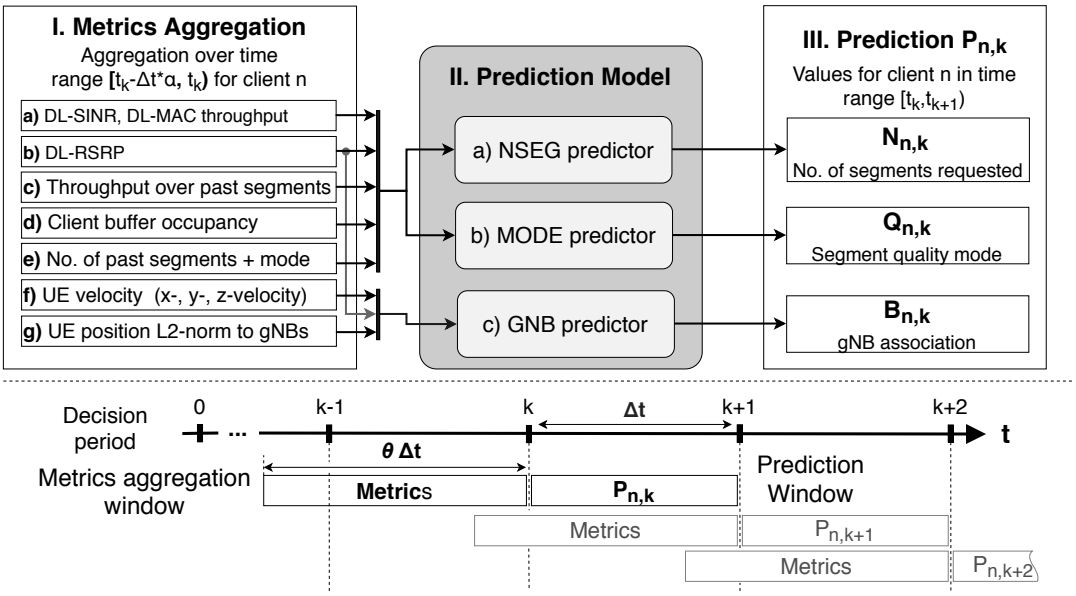


FIGURE 4.3: Prediction inputs and outputs at a decision instant.

on the requests and responses that pass through the MEC. The overall complexity of the prediction procedure scales linearly with the number of users in the network.

4.4.2 Prefetching Models

Mobile Network Model. Let $G = (N, E)$ be an undirected graph modeling the mobile network, where N represents the computing nodes, which are the union of the set of gNBs N_{gnb} and the 5GC N_{5gc}, $N = N_{gnb} \cup N_{5gc}$. E represents the set of backhaul and Xn links, interconnecting the gNBs with the 5GC and gNBs with each other, respectively. As already mentioned, each node $n \in N$ has a collocated MEC server that is characterized with a storage $C_{stg}(n)$ and processing capacity $C_{cpu}(n)$. While the former is used to cache video segments, the latter, if needed, is used to transcode video segments from a high bitrate h to a lower bitrate q, which is the one predicted to be requested by the UE. There is a link $e^{m,n} \in E$ between the nodes $m, n \in N$ if they are directly connected, which has a certain amount of bandwidth denoted by C^e_{bwt}. N_{vid} represents the set of videos available to the UEs. Each video $v \in N_{vid}$ is divided into multiple segments N^v_{seg}, each of which $s \in N^v_{seg}$ is available in multiple bitrates $N^{v,s}_{br}$. Table 4.1 summarizes the parameters of the mobile network.

UE Request Model. The UE requests are modeled as a directed graph $\bar{G} = (\bar{N}, \bar{E})$, where $\bar{N}$ is the union of UEs and their requested bitrate of a specific video and segment, $\bar{N} = \bar{N}_{ue} \cup \bar{N}^{v,s}_{br}$, and $\bar{E}$ represents the virtual links between UEs and their requested bitrate. Moreover, $\omega_{br}(r)$ represents the bitrate of the given requested

TABLE 4.1: Mobile network parameters

Parameters	Description
$G = (N, E)$	Graph representing the mobile network.
N	Set of nodes that can host videos $N = N_{gnb} \cup N_{5gc}$.
E	Set of links connecting the nodes in G.
N_{gnb}	Set of gNBs in G.
N_{5gc}	Set of core nodes/servers in G.
N_{vid}	Set of videos.
N_{seg}^{v}	Set of segments of each video $v \in N_{vid}$.
$N_{br}^{v,s}$	Set of available bitrates for each segment $s \in N_{seg}^{v}$ of video $v \in N_{vid}$. Bitrates are in order from the lowest to the highest $q_1 < ... < q_5$.
$\omega_{cpu}^{h,q}$	Number of CPU cores required for transacting a segment from bitrate h to the desired bitrate q of the user.
$C_{cpu}(n)$	Number of CPU cores available on node $n \in N$.
$C_{stg}(n)$	Caching storage of node $n \in N$ in Megabytes.
$C_{bwt}(e)$	The bandwidth capacity of the substrate link $e \in E$.
τ^{s}	Segment time duration. All the segments are considered to be in the same duration.
α	The weight factor to prioritize the embedding options.

video segment by the UE. It is possible to have multiple requests from the same UE in any given time. Table 4.2 summarizes the notations used for the service requests.

Problem Formulation. The joint video segment pre-fetching, transcoding, and resource allocation problem is modeled as a VNE problem, which is proven to be NP-hard [18]. The embedding process is performed in two steps, including the node embedding and the link embedding step. In the node embedding step, each virtual node (e.g., UEs and video segments) in the request is mapped to a substrate node (e.g., gNBs). In the link embedding instead, each virtual link is mapped to a single substrate path.

ILP-based Method

ILP techniques are employed to formulate the described VNE problem that has two objective functions. While the first objective (4.1) tends to maximize the cache-hit ratio, the second (4.2) maximizes the byte-hit ratio. Table 4.3 summarizes the variables used in the ILP model.

TABLE 4.2: UE request parameters

Parameters	Description
$\bar{G}(\bar{N},\bar{E})$	Video request graph.
$\bar{N}$	Set of requests in $\bar{G}$.
$\bar{N}^{r}_{vid}$	The video in the request $r \in \bar{N}$.
$\bar{N}^{r,v}_{seg}$	The segment of video $v \in N_{vid}$ in the request $r \in \bar{N}$.
N^{r}_{br}	The bitrate of the request r for its video segment.
$\omega_{br}(r)$	The bitrate of a video segment for the UE's request $r \in \bar{N}$.
$\bar{E}$	Set of links connecting UEs to the requested bitrate in $\bar{G}$.

There are multiple ways to increase the cache-hit ratio. The UE requested video segment(s) with a specific bitrate, if already cached/available, can be served either from the host gNB or from a neighbor gNB leveraging the Xn interface. If the requested bitrate of the desired segment is not already available at any of the gNBs/edge sites while higher bitrates of the same video segments are available, then they can be transcoded to the desired bitrate/quality. Note that the video segment prefetching is based on the prediction of the UE-requested video segment along with its bitrate for the subsequent timeslot.

$$CacheHit: \quad max \sum_{r\in\bar{N}} \sum_{n\in N_{gnb}} \sum_{\substack{h\in N^{r}_{br} \\ h\geq q}} \alpha \chi^{r,h}_{n} \tag{4.1}$$

The second objective function (4.2) aims to maximize the byte-hit ratio, employing the same weighting factors. This objective is particularly beneficial for storing videos with high storage demand at the edge, resulting in backhaul load alleviation.

$$ByteHit: \quad max \sum_{r\in\bar{N}} \sum_{n\in N_{gnb}} \sum_{\substack{h\in N_{br}(r) \\ h\geq q}} \alpha \omega_{br}(r) \chi^{r,h}_{n} \tag{4.2}$$

In the following, we present the constraints that, regardless of the objective function, have to be satisfied for a valid solution. The first constraint ensures that the storage used for storing the videos is less than or equal to the maximum storage capacity of the edge nodes.

$$\forall n \in N_{gnb}: \sum_{v\in N_{vid}} \sum_{s\in N^{v}_{seg}} \sum_{q\in N^{v,s}_{br}} \omega^{v,s}_{br} \tau^{s} \chi^{v,s,q}_{n} \leq C_{stg}(n) \tag{4.3}$$

At any given time, each UE should be provided with one video segment in the requested bitrate.

$$\forall r \in \bar{N} : \sum_{n \in N} \sum_{\substack{h \in N_{br}^{r} \\ h \geq q}} \chi_{n}^{r,h} = 1 \tag{4.4}$$

The following constraint guarantees that a video segment is available at the edge only if at least one UE is requesting that video segment.

$$\forall n \in N, \forall v \in N_{vid}, \forall s \in N_{seg}^{v}, h \in N_{br}^{v,s} : \sum_{r \in \bar{N}} \chi_{n}^{r,h} - \mu \chi_{n}^{v,s,h} \leq 0 \tag{4.5}$$

where s and v are the segment and the video of request r, while μ is a big number. Constraint (4.6) ensures that the virtual links are mapped on a substrate link as long as the link has sufficient capacity.

$$\forall e \in E : \sum_{\bar{e} \in \bar{E}} \omega_{br}(\bar{e}) \chi_{e}^{\bar{e}} \leq C_{bwt}(e) \tag{4.6}$$

Constraint (4.7) enforces a continuous path established between the UE and the bitrate of the video segment in the virtual request $r \in \bar{N}$.

$$\begin{aligned} &\forall i \in N, \forall e^{m,n} \in \bar{E} : \\ &\sum_{e \in E^{i\rightarrow}} \chi_{e}^{e^{m,n}} - \sum_{e \in E^{\rightarrow i}} \chi_{e}^{e^{m,n}} = \begin{cases} -1 & \text{if } i = m \\ 1 & \text{if } i = n \\ 0 & \text{otherwise,} \end{cases} \end{aligned} \tag{4.7}$$

where $E^{i\rightarrow}$ represents the links originating from node $i \in N$, while $E^{\rightarrow i}$ represents all the links entering node $i \in N$.

Finally, constraint (4.8) makes sure that the number of CPU cores utilized for transcoding a video segment in bitrate $h \in N_{br}^{v,s}$ to the lower desired bitrate $q \in \bar{N}_{br}^{r}$ are not higher than the number of CPU cores available at the edge.

$$\forall n \in N_{gnb} : \sum_{r \in \bar{N}} \sum_{\substack{h \in N_{br}^{r} \\ h > q}} \omega_{cpu}^{h,q} \chi_{n}^{r,h} \leq C_{cpu}(n) \tag{4.8}$$

Heuristic

Although the ILP model achieves the optimal solution in different network configurations, the problem becomes computationally intractable with the increase in the

TABLE 4.3: Binary decision variables

Variables	Description
$\chi_n^{v,s,h}$	Indicates if the bitrate $h \in N_{br}^{v,s}$ of the segment $s \in N_{seg}^{v}$ of video $v \in N_{vid}$ has been mapped on the edge node $n \in N_{gnb}$.
$\chi_n^{r,h}$	Indicates if the request $r \in \bar{N}$ of the video segment bitrate h has been mapped on the node $n \in N$.
$\chi_e^{\bar{e}}$	Indicates if the virtual link $\bar{e} \in \bar{E}$ is mapped on the substrate link $e \in E$.

network size, the number of UEs, videos, and their segments. In order to tackle this issue, we propose a heuristic algorithm (see Algorithm 1) that is able to reach a near-optimal solution for video segment prefetching, transcoding, and resource allocation in a very limited time scale even for very complex network configurations.

The proposed algorithm, named *Heu cache_hit*, follows the same objective of maximizing cache-hit ratio for the video segments. Like the ILP cache-hit algorithm, *Heu cache_hit* needs to have the weighting factors before running the algorithm. For each request $r \in \bar{N}$, the weighting factor α is computed for each node $n \in N$ by taking into consideration the UE-gNB association and the desired state of the MNO, which indicates the preference for mapping the request. For example, the MNO may give the highest preference for serving the UE requests from the same gNB that the UEs are associated. The next preferred option would be to choose transcoding a higher bitrate to the desired bitrate or use the neighbor gNBs to retrieve the segment bitrate. Serving the UEs from the 5GC gets the least priority.

The algorithm considers each request predicted to be issued during the next time window, extracting its predicted bitrate q and the host gNB g. We remind the reader that the video, segment, and bitrate are inherently included in the request. The heuristic then calculates the weighting factors α for each request based on the UE-gNB association and sorts them in descending order.

This is followed by traversing all the bitrates and nodes as possible solutions in the sorted list of the weighting factors α. The $alloc$ is an array to store the embedding decisions. If the candidate solution has already been embedded on the substrate network, indicated by $alloc[r,h,n] == 1$, then the UE will be served from the same video segment on the same node without prefetching a new segment. If the desired content is on a node different from the host gNB, then a path will be established between the host gNB and the node serving the desired content if the physical links in the path have sufficient capacity.

Algorithm 1: *Heu cache_hit*

Input: $(G, \bar{G})$
Output: (video, segment, bitrate) pre-fetching, transcoding, and resource allocation ;

1 **for** $r \in \bar{N}$ **do**
2 $q \leftarrow \bar{N}^r_{br}$;
3 $g \leftarrow gnbAssociation(r)$;
4 • Compute α weighting factor for request r;
5 • Sorte α in descending order;
6 **for** $h, n \in \alpha$ **do**
7 **if** $alloc[r,h,n] == 1$ **then**
8 **if** $n \neq g$ ***and*** $C^{(n,g)}_{bwt}(e) > \omega_{br}(r)$ **then**
9 • Allocate path $P_{n,g}$ and update resources;
10 • Allocate (h,n,r) and update resources;
11 **break**;
12 **else**
13 **if** $C_{stg}(n) > \omega^r_{br}(n) * \tau^s$ **then**
14 **if** $h > q$ ***and*** $\omega^{h,q}_{cpu} > C_{cpu}(n)$ **then**
15 **if** $n \neq g$ ***and*** $C^{(n,g)}_{bwt}(e) > \omega_{br}(r)$ **then**
16 • Allocate path $P_{n,g}$ and update resources;
17 • Allocate (h,n,r) and update resources;
18 $\omega^{h,q}_{cpu}(n) \leftarrow \omega^{h,q}_{cpu}(n) - \omega^{h,q}_{cpu}$;
19 **break**;
20 **else**
21 **if** $n \neq g$ ***and*** $C^{(n,g)}_{bwt}(e) > \omega_{br}(r)$ **then**
22 • Allocate path $P_{n,g}$ and update resources;
23 • Allocate (h,n,r) and update resources;
24 **break**;

In the case of the absence of the solution on the node, if the node has enough storage capacity, the heuristic will check if the bitrate of the solution needs transcoding to be performed on it. If transcoding is required (line 14), the solution will be assigned to the request. The CPU capacity on the node will be deducted; otherwise, the solution will be embedded, and the paths will be allocated at lines 22 and 23. In general, the algorithm's complexity grows linearly and depends on the network size, including the number of edge nodes, number of videos, number of segments, and number of available bitrates.

4.5 Performance Evaluation

This section provides an in-depth comparison between the proposed ILP and heuristic algorithms with the baseline based on the performance results derived from the predictor models.

4.5.1 Data Collection for Prediction and Evaluation

An ns-3 [99] simulated network is used to generate data to train the prediction algorithms and to evaluate the pre-fetching ILP and heuristic algorithms. The evaluation of the ILP and heuristic evaluation is done offline by processing the output from the predictor and the ground-truth from the ns-3 generated data. Two separate datasets [100] have been generated from a simulated urban mobile network deployment scenario with the same setup parameters. We use the ns-3 DASH module implementation by Vergados et al. [101] to simulate the DASH client-server interaction of segment requests and response.

The simulation setup consists of 12 gNBs and uses carrier aggregation with three component carriers of 20 Mhz to provide a maximum downlink bandwidth of 60 Mhz, which can reach an aggregated downlink bitrate of up to 225 Mbps. A variable number of UEs (between 27 - 68) move between these gNBs with velocities ranging between 1.4 - 5.0 m/s (walking/cycling speeds expected in the considered deployment scenario of a large arena). UEs, use DASH to stream video content in this dynamic wireless network environment. Data is collected over 27 runs of 1000 seconds each.

The UEs in the network request segments from one of the 10 videos that are currently being streamed. The UE's video stream play times are within 10 seconds of each other, representing realistic behavior of live streams that are viewed within small delays of the actual event (this delay could be due to replays or delayed requests). The videos are streamed at 50 frames per second, with segments' duration of 8 seconds. The set of available bitrates are $\{1, 2.5, 5, 8, 16, 35\}$ Mpbs (higher rates correspond to HD, 2K, and 4K video qualities).

4.5.2 Prediction

A total of 27 runs of data (each 1000 seconds of simulation) are used: 23 of them for training the predictors, and 4 for the evaluation of predictive pre-fetching as described

in in Sec. 4.5.3. We employ Random Forest (RF) for the predictor models since it has the flexibility of a non-parametric method and has been demonstrated to perform well for similar tasks [54], [56]. Additionally, we explore Gradient Boosting Trees (XGB) for classification [102] due to prior evidence of boosting trees outperforming RF [103].

Pre-Processing. The data pre-processing step consists of reading and parsing the log files generated by various network monitoring components. Only requests *after* the first requested segment are considered (the task is not to predict when the client starts requesting). The prediction window is then slid through each client's trace data until the last segment is requested. To get the input features, the data is first divided according to client-id. The required metrics are then extracted over the metrics aggregation window for each prediction interval. All the metrics and ground truths are compiled into one data structure (one for each client).

Prediction Window Size. This parameter directly influences the number of possible classes for the NSEG- and MODE-predictor. Therefore, the class occurrence distribution is computed for varying window sizes in Fig. 4.4, according to Sec. 4.5.2. While there is a small influence on the MODE class distribution, there is a significant influence on the NSEG class distribution. Since one bitrate is predicted for all segments in the prediction window, the ideal implies having one segment in each window. This occurs with a 2-sec window size but results in an imbalanced class distribution (no segment requested $\sim 80\%$ of the time). Additionally, the window size must be significantly larger than both the ILP and heuristic running time. For these reasons, we select window sizes of 4, 8, and 16 sec to find the best performing predictors. The prediction window size determines the number of training samples: 211630, 105484, and 52010 samples are generated for respectively 4, 8, and 16-sec window sizes.

Training. We employ a train-test set split of 80%-20%. The XGB models use the softmax objective with the default learning-rate and regularization [102]. The RFs perform split decisions based on the Gini impurity, and trees are built using bootstrapping. For both RF and XGB we perform a grid search with the following hyperparameters: i) the number of estimators (number of trees) with possible values $\{5, 15, 50, 100, 200, 350\}$, and ii) the maximum tree depth with possible values $\{5, 15, 25\}$. It results in a total of 36 models to train for each predictor. We use 10-fold cross-validation during training, and the best performing model configuration in terms of accuracy on the test set is saved.

Model Performance. Table 4.4 summarizes the models' performance. The

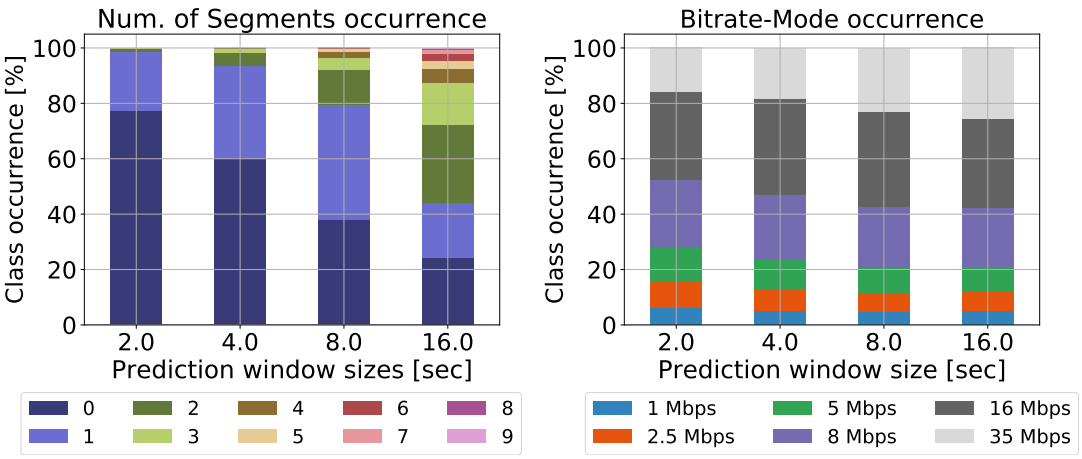


FIGURE 4.4: Prediction window size analysis

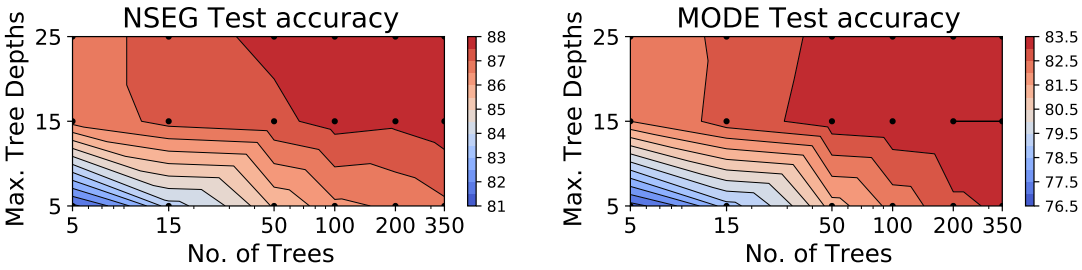


FIGURE 4.5: Train-Test accuracy hyperparameter dependency.

best NSEG-predictor (test accuracy of 83.3%) employs XGB with 350 estimators and 25 maximum depth using 4-sec for both prediction and metrics aggregation windows. The best MODE-predictor (test accuracy of 89.7%) employs XGB with 350 estimators and 25 maximum depth using a 4-sec prediction and a 10-sec metrics aggregation window size. Overall, XGB is generally capable of outperforming RF regardless of the window sizes combination due to its capability of reducing variance error. Moreover, irrespective of the prediction window size, an increase of the metrics aggregation window size decreases the NSEG model's performance. An increase in the prediction window size also reduces overall performance: even matching the number of training samples of the 4-sec prediction window size to that of the 16-sec window yields a 80.7% test accuracy. Furthermore, Fig. 4.5 shows how increasing tree depth and the number of estimators positively affects the test accuracy significantly until reaching 100 trees and a depth of 15 and 5 for NSEG and MODE, respectively. Since the GNB-predictor problem is fairly simple due to the smaller number of metrics with a simple relationship to the target prediction, it obtained a test accuracy close to 100% regardless of the model employed. Overall, the high accuracy score on the training sets suggests that the models are able to learn the separation hyperplanes to classify the data, but the corresponding test accuracy suggests that

the models generally overfit, even with the depth of 5. The best performing models are used to compute the ILPs input.

4.5.3 Prefetching

This section elaborates on the simulations carried out in Python using the Gurobi mathematical optimization solver [15]. The ILP and heuristic evaluation considers the following additional parameters during the offline processing of data. The 5GC, where the DASH video server runs, is equipped with a 16-core 2.4 GHz processor. The edge MEC server is equipped with a 4-core 1.6 GHz processor with varying storage of $\{25, 50, 75, 100, 125, 150\}$ MB.

This storage space is shared between all applications supported by the edge, requiring efficient caching strategies. The edge MEC servers are interconnected over 5 Gbps Xn links, which, along with exchanging control information, are also used to transfer video segment data. The MEC servers are connected to the 5GC over a 20 Gbps backhaul link.

The prefetching algorithms (ILP or heuristic) run periodically every four seconds, in what we call a prefetching time slot. In each prefetching time slot, the Decision Maker at the 5GC obtains the segment/bitrate/association predictions for all UEs from the Predictor component at the MEC and puts together a set of prefetching requests. Each of the quadruple (UE, segment, bitrate, gNB) is considered as a request. The prefetching algorithm tries to find a solution that can serve all predicted requests. The ILP and the faster heuristic still take a finite amount of time to run. However, the prefetching time slot is always longer than the time to run the prefetching algorithm to ensure that the requests from one slot are fetched before the beginning of the next.

The predicted requests can be served by either prefetching the requested segments to the edge, transcoding an existing segment at the edge, or skipping the cache and waiting for the 5GC to serve the actual request. Three cases result in a user being served from the 5GC. (i) the ILP decides that the user should be served from the core, (ii) due to error in prediction, the wrong segment bitrate was prefetched (iii) due to error in gNB association prediction, the prefetched segment was placed at the wrong edge MEC server. The cache at the MEC server uses a simple Least Recently Used (LRU) replacement strategy for content management. The Cache Manager informs the Request Handler about the cache status and the requests that

TABLE 4.4: Accuracy of the trained ML models using different prediction window and metrics aggregation window sizes.

Prediction w-size:		4 sec						8 sec						16 sec			
Metrics w-size:		4 sec		10 sec		20 sec		8 sec		10 sec		20 sec		16 sec		20 sec	
Acc. for set [%]:		Train	Test	Train	Test	Train	Test	Train	Test	Train	Test	Train	Test	Train	Test	Train	Test
NSEG	RF	97.4	82.8	97.6	79.5	97.3	77.0	96.5	75.2	96.5	74.7	96.1	73.4	95.8	**60.2**	95.8	58.3
	XGB	98.3	**83.3**	98.1	80.7	86.4	79.3	97.6	**76.0**	97.6	75.5	97.4	73.9	95.9	59.5	95.9	57.9
MODE	RF	98.6	87.1	98.8	88.8	98.6	86.3	98.5	85.5	98.5	85.0	98.1	81.9	97.7	77.4	97.5	76.3
	XGB	98.8	88.0	98.9	**89.7**	98.7	87.8	98.6	86.0	98.6	**86.3**	98.3	83.3	97.7	**77.6**	97.6	77.2

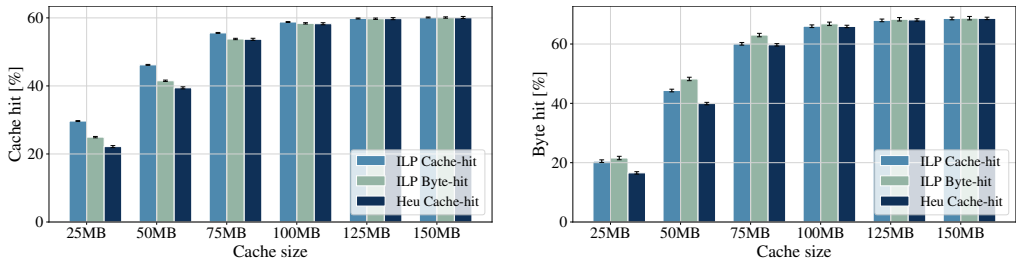
can be served from the edge, and it frees up cache space if needed by discarding any other segments based on LRU.

The reported results are the average of four simulation runs with 95% confidence intervals. During each simulation run, a variable number of requests will be issued, and the prefetching algorithm is responsible for making an intelligent decision to embed the requests on the substrate network.

Cache-hit Ratio. As stated earlier, a request can be served from the edge in multiple ways. The first solution is to serve the request directly from the gNB that the UE is associated with, and it can be either by the exact bitrate stored at the MEC node or by transcoding a higher bitrate segment to the target bitrate. The second solution is to serve the request through the neighbor gNBs with the same methods. We define the cache-hit ratio as the ratio between the number of requests served from the edge to the number overall number of requests. We argue that the higher cache-hit results in higher satisfaction for both the end-users and the MNO. While the end-users experience less delay, less jitter, and higher bitrate, the MNOs can offload the backhaul to a large extent. Here we study the cache-hit ratio under different cache sizes. It is clear that increasing cache size results in higher cache-hit. Therefore, we want to study the impact of the cache-hit ratio in different network configurations. It is worth mentioning that we decrease the cache size to a large extent to show the system's different behaviors for testing purposes.

As shown in Fig. 4.6a, the average cache-hit ratio with different cache sizes is higher for the ILP cache-hit algorithm compared to the other two algorithms. As was expected, the ILP cache-hit and Heu cache-hit achieve, respectively, the first and the second highest cache-hit ratio due to the importance of the number of hits in the objective function. The superior performance of the ILP cache-hit becomes obvious when the cache storage is very limited, and all the predicted segments cannot be accommodated at the edge MECs. Therefore, this is the scenario where ILP cache-hit can make more intelligent decisions compared to ILP byte-hit and Heu cache-hit algorithms in selecting a set of segments and bitrates to be prefetched that leads to the maximum cache hit. Although, with the increase in the cache size, we encounter a scenario in which all the predicted segments can be stored at the edge nodes, therefore, making the prefetching decision straightforward for all of the algorithms to prefetch whatever predicted and achieving a high cache-hit ratio.

Byte-hit Ratio. Similar to the cache-hit ratio, we also study the byte-hit rate of the proposed algorithms. Obviously, with the more bytes served at the edge, the more savings over the backhaul link can be archived. We intended to depict the ratio



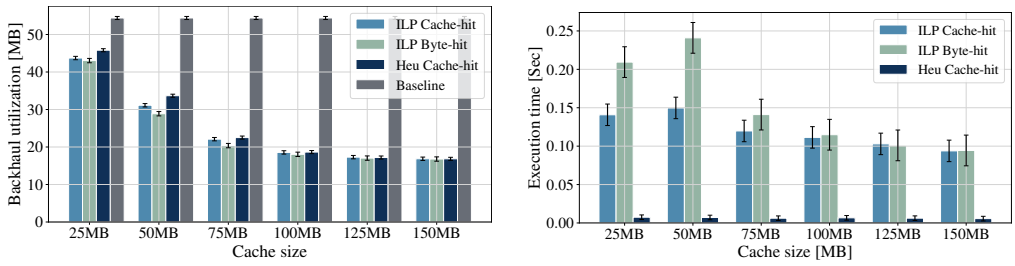
(A) Cache-hit ratio for different cache sizes. (B) Byte-hit ratio for different cache sizes.

FIGURE 4.6: Average cache-hit ratio, byte-hit ratio and backhaul link utilization for different cache sizes.

between the number of bytes served from the edge and the overall number of bytes requested with this performance metric. Therefore, we expect the ILP byte-hit to perform better than the other algorithms in prefetching higher bitrate segments that can be shared among multiple UEs to the edge to save the bandwidth. Another advantage with the prefetching of high-bitrate segments is that a higher bitrate segment can be transcoded to the lower bitrate segments and avoid of re-directing requests for low-bitrate segments to the core and serve more users from the edge.

As illustrated in Fig. 4.6b, the number of bytes served from the edge for the ILP byte-hit algorithm is more than the other two algorithms. Similar to the cache-hit evaluation, the algorithm shows a better comparable performance concerning the other algorithms when the cache size is smaller, and the algorithm needs to decide on prefetching intelligently video segments to the edge. Here, the ILP cache-hit shows a close performance to ILP byte-hit because it can prefetch more number of segments, increasing the number of served bytes, but still being less than the ones archived by ILP byte-hit. As expected, here also, the Heu cache-hit achieves a near performance to what achieved by ILP cache-hit.

Link Utilization. Figure 4.7a illustrates the backhaul link utilization as a function of cache storage size, averaged over four simulation runs and compared with the baseline in which all the segments are served from the core, and no pre-fetching is performed. We can observe that Heu cache-hit and ILP byte-hit achieve, respectively, the highest and the lowest backhaul H link utilization compared with the baseline. This is justified by the fact that ILP byte-hit tends to pre-fetch high bitrate segments that might be shared among multiple requests to the edge. Therefore, a smaller portion of the high bitrate requests will be directed to the core, and the backhaul link will be saved. As can be understood from Fig. 4.7a, the proposed algorithm can save up to 69.15% of the backhaul link compared to the baseline. This happens when the cache size is large but still performs better than the other algorithms, even with a very limited cache size. Conversely, the Heu cache-hit algorithm follows the



(A) backhaul link utilization for different cache sizes.

(B) Execution time for different cache sizes.

FIGURE 4.7: Average backhaul link utilization for different cache sizes and execution time.

same objective of the ILP cache-hit but shows a lower performance compared to the ILP-based algorithms.

Execution Time. The main goal of the proposed Heuristic algorithm (Heu cache-hit) is to combat the scalability issue of the ILP cache-hit algorithms, which become computationally intractable with the increase in the network size, the number of videos, their segments, and the number of UEs requesting those video segments. Fig. 4.7b demonstrates the average execution time of the proposed algorithms over four runs. It can be observed that the execution time of the ILP algorithms is significantly higher than that of the Heuristic. Thus, the Heu cache-hit, due to its sub-optimal mapping solutions, exhibits lower performance compared to its ILP counterpart in terms of cache-hit ratio, byte-hit ratio, and link utilization, it proves to be competitive and also applicable to extensive size networks in real-world scenarios.

4.6 Discussion

In this chapter, we proposed a novel method for ML-driven predictive prefetching for the problem of DASH video streaming in MEC-enabled mobile networks. We showed that with an accuracy of (83-88-99%) for the three predictive tasks, we are able to attain a MEC cache-hit ratio of 60%, which means that we were able to reduce the access delay for 60% of the requests. The ML algorithm predicts the number of segment requests, bitrate, and the gNB association of the UEs in a prediction time window. An ILP model with two objectives was proposed to reach an optimal solution for the video content prefetching and transcoding at the edge, followed by a heuristic algorithm that achieves a near-optimal solution in an incredibly shorter time scale. It is demonstrated that the backhaul link utilization can be reduced by 69.15% through caching at the edge using max byte-hit objective in a live streaming

scenario with segment request overlaps. The developed Heuristic resulted in the reduction of the execution time for prefetching in the 25 MB cache size scenario by 90% with a reduction in the cache-hit ratio by 7.5% and an increase in the backhaul link utilization of 2.08%.

Chapter 5

Latency-Aware User Association and SFC Placement

This chapter uncovers the motivations towards latency-aware user association and SFC placement in MEC-enabled 5G networks. The main objective of the chapter is to propose novel user association and SFC placement algorithms to find optimal gNBs and computing nodes to embed users' requested services. The user association and SFC placement problems are formulated employing MILP techniques, having the objectives to minimize service provisioning cost, transport network utilization, and the impact of VNF migration on users' experienced QoE. A heuristic algorithm is also proposed supporting the idea of minimizing the number of users affected by VNF migration in a considerably shorter time scale. Comprehensive numerical experiments are performed to draw a comparison between these approaches.

5.1 Overview

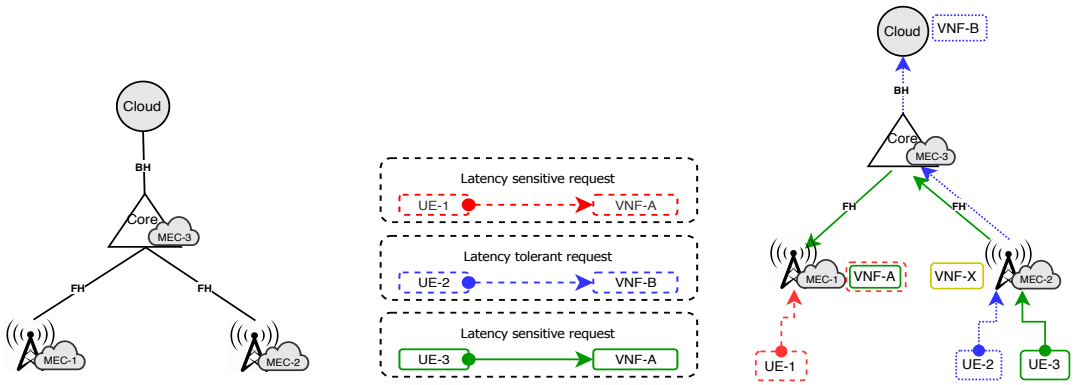
The 5th generation (5G) of cellular networks undertakes the mobile communication landscape transformation by offering an extremely high QoE (e.g., low-latency, high data rate) for the end-users. MEC [104] is a key enabler in the 5G network by shifting the applications, services, and processing capabilities closer to the end-users and, therefore, offloading the transport network reducing the round-trip delay experienced by the end-users. MEC servers may reside along with the gNBs as well as with the core network. While these MEC servers can be used to host low-latency VNF applications, the cloud data centers can be used to accommodate the latency-tolerant ones. In general, the closer the MEC server is located to the user,

the less is its computational capacity, which means that the more costly is VNF instantiation on that MEC server. Given the above considerations and the number of users requesting various applications with diverse QoS requirements, the natural question that arises is which gNBs to associate the users to and where to deploy their requested applications, such as to make sure that their application requirements are satisfied while the network resources are used most efficiently.

This chapter provides a comprehensive E2E delay estimation model for users, taking into account the transmission and propagation time over the air and the transport links along with the VNF processing time. We employ MILP techniques to provide a novel formulation of the joint user association, SFC placement, and resource allocation problem. Aiming at alleviating the scalability issues of the proposed MILP formulation, a heuristic algorithm that reaches a near-optimal solution is proposed to minimize the impact of VNF migration on user QoE in a much shorter time scale. We perform comprehensive simulations, drawing a comparison between the proposed algorithms by considering different types of service requests with diverse data rates and E2E latency requirements.

5.2 Problem Statement

Figure 5.1a depicts the reference network architecture in which the gNBs are collocated with MEC servers, referred to as edge nodes, and are in charge of providing coverage to the users and performing their baseband signal processing. The edge nodes have a limited amount of computational capacity, which makes their usage quite costly. It is essential to mention that we also consider the case in which a user can be associated with one gNB while be served by a MEC server collocated with another gNB. While all the nodes possess computing capabilities, only the gNBs and the core are equipped with MEC servers. As opposed to the gNBs, the MEC server collocated with the core node has much more computational capacity, making the VNF instantiation much cheaper. Nevertheless, VNF instantiation on the core node requires the use of the fronthaul transport resources, which contributes to the total cost computation for the VNF instantiation. As for the cloud data center, it has abundant computational resources, which makes it the cheapest solution to be used for instantiating VNFs compared to the edge nodes and the core, regardless of the additionally required transport network resources (i.e., both fronthaul and backhaul resources). Thus, the closer is the computing node to the end-user, the less is its computation capacity.



(A) Mobile network. (B) Service request. (C) VNF Mapping.

FIGURE 5.1: Sample mobile network and service request models.

It is assumed that each user or UE requests a service with a certain bitrate and delay tolerance. Upon receiving the service request from the UE, the network provider shall decide on how to associate the UE to the network and embed his request, such as to make sure that the UE service requirements are satisfied, while the network resources are used in the most efficient manner. Figure 5.1b depicts the service requests composed of UEs and the requested service, having either strict latency or loose latency requirements, which are numerically defined in Section 5.4.1. Figure 5.1c illustrates a sample service mapping whose objective is to minimize the service provisioning cost. The service requested by UE-2 is placed in the cloud, while the services of UE-1 and UE-3 are mapped on the MEC-1 server at the edge due to their strict latency requirement. Note that since the VNF requested by UE-3 is already available on MEC-1, UE-3 is served by that VNF in order to reduce the service provisioning cost while satisfying the latency demand.

The problem of joint user association, SFC placement, and resource allocation can be formally stated as follows.

Given: a 5G network composed of gNBs and a core node that have collocated MEC servers and are interconnected via fronthaul links. Additionally, given a cloud data center node that is interconnected with the core node via a backhaul link. Moreover, given a set of UEs randomly scattered in a geographical area, each requesting a service with its respective data rate and latency requirement.

Find: joint user association, SFC placement, and resource allocation.

Objective: minimize (i) the service provisioning cost, (ii) the impact of VNF migration on UEs' QoE, and (iii) the transport bandwidth consumption.

The joint user association, SFC placement, and resource allocation problem is modeled as a VNE problem and has been studied extensively in the literature [105], [106]. The embedding process consists of two phases: the node embedding and the link embedding. In the node embedding phase, each virtual node (e.g., UEs and VNFs) in the request is mapped to a substrate node (e.g., gNBs, core servers, and cloud nodes in the substrate network). In the link embedding instead, each virtual link is mapped to a single substrate path. In both cases, constraints of nodes and links must be satisfied in order for a solution to be valid.

5.3 Proposed Methods

Mobile Network Model. Let $G = (N, E)$ be an undirected graph modeling the mobile network, where N represents the computing nodes, which are the union of the set of gNBs N_{gnb}, the core N_{core}, and the cloud N_{cloud}, $N = N_{gnb} \cup N_{core} \cup N_{cloud}$. E represents the set of fronthaul and backhaul links interconnecting, respectively, the gNBs with the core and the core with the cloud. Each computing node $n \in N$ in the network is equipped with a certain amount of processing capacity represented by $C_{cpu}(n)$. There is a link $e^{m,n} \in E$ between the nodes $m, n \in N$ if they are directly connected.

Let ω^i_{cpu} represent the number of CPU cores assigned to the instance $i \in N^s_{inst}$ of service $s \in N_{vnf}$, which is represented as a single VNF. It is assumed that at least a single CPU core is required to spawn/instantiate a VNF, while it is also possible to allocate three CPUs to a VNF instance depending on the data processing demand. $C^i_{proc}(n)$ is the processing capacity of instance $i \in N^s_{inst}$ of VNF $s \in N_{vnf}$ on node $n \in N$. There is an upper bound on the number of UEs that can use a single CPU core. Thus, the capacity of a VNF instance $C^i_{ue}(n)$ can be expressed in terms of the maximum number of UEs that can use that VNF, which depends on the number of CPU cores allocated to that VNF. It is worth to mention that we also tackle the case in which multiple instances of the same VNF are needed due to high traffic demand. Finally, each link $e^{m,n} \in E$ connecting the nodes $m, n \in N$ in the network has a certain bandwidth capacity $C_{bwt}(e)$ in Gbps. Table 5.1 summarizes the parameters of the mobile network.

UE Request Model. We model the service requests as a directed graph $\bar{G} = (\bar{N}, \bar{E})$, where $\bar{N}$ is the union of UEs and their requested services, $\bar{N} = \bar{N}_{ue} \cup \bar{N}_{vnf}$, and $\bar{E}$ represents the virtual links between UEs and their requested services. It is assumed that

TABLE 5.1: Mobile network parameters.

Parameters	Description
$G(N,E)$	Graph representing the mobile network.
N_{gnb}	Set of gNBs in G.
N_{core}	Set of core servers in G.
N_{cloud}	Set of cloud servers in G.
N	Set of computing nodes in G.
E	Set of links connecting the nodes in G.
N_{vnf}	Set of services.
N^s_{inst}	Set of instances of service $s \in N_{vnf}$.
$N^{i\star}_{ue}(n)$	The number of UEs $u \in \bar{N}_{ue}$ served from the service instance $i \in N^s_{inst}$ on the node $n \in N$ in the previous run.
ω^i_{cpu}	The number of CPU cores that are required to run instance $i \in N^s_{inst}$ of service $s \in N_{vnf}$.
ω^g_{prb}	The amount of PRB available on gNB $g \in N_{gnb}$.
ξ^n_{cpu}	The cost of one CPU core on node $n \in N$.
ξ^e_{bwt}	The cost of using one Mbps bandwidth of link $e \in E$.
ξ^g_{prb}	The cost of using one PRB in gNB $g \in N_{gnb}$.
C^u_g	The maximum achievable data rate between UE $u \in \bar{N}_{ue}$ and gNB $g \in N_{gnb}$.
$C^i_{ue}(n)$	The maximum number of UEs that can use the instance $i \in N^s_{inst}$ of service $s \in N_{vnf}$ on node $n \in N$.
$C_{cpu}(n)$	The CPU cores of node $n \in N$.
$C^i_{proc}(n)$	Processing capacity of instance $i \in N^s_{inst}$ of service $s \in N_{vnf}$ on node $n \in N$.
$C_{bwt}(e)$	The bandwidth capacity of the substrate link $e \in E$.
$d_{(g,u)}$	Distance between gNB $g \in N_{gnb}$ and UE $u \in \bar{N}_{ue}$.
P^g_{tx}	The transmission power of gNB $g \in N_{gnb}$.
μ	A big positive number.

UEs are randomly scattered in a geographical area, and each UE can be associated to only one gNB.

In our model each UE $u \in \bar{N}_{ue}$ requests only one service $s \in \bar{N}_{vnf}$, specifying the maximum delay tolerance by T^u_{max} and data rate demand ω^u_{bwt}. The allocated VNF instance should process the data transmitted by the UE. The total delay of the service is calculated as the summation of the transmission time over the air, which is considered to be equal to one transmission time interval (TTI = 1ms), transmission time over fronthaul and backhaul links, propagation time over the air,

TABLE 5.2: UE request parameters.

Parameters	Description
$\bar{G}(\bar{N},\bar{E})$	Service request graph.
$\bar{N}$	Set of UEs and requested services $\bar{N} = \bar{N}_{ue} \cup \bar{N}_{vnf}$ in $\bar{G}$.
$\bar{N}_{ue}$	Set of UEs in $\bar{G}$.
$\bar{N}_{vnf}$	Set of services requested by the UEs in $\bar{G}$.
$\bar{E}$	Set of virtual links connecting UEs to the services in $\bar{G}$.
ω^{u}_{bwt}	Data rate requested from UE $u \in \bar{N}_{ue}$.
$\omega^{u,g}_{prb}$	The number of required PRBs to support the data request of UE $u \in \bar{N}_{ue}$ from gNB $g \in N_{gnb}$.
T^{u}_{max}	Maximum delay tolerance of UE $u \in \bar{N}_{ue}$.
$T^{u}_{tx}(g)$	The transmission time between UE $u \in \bar{N}_{ue}$ and gNB $g \in N_{gnb}$.
$T^{u}_{prp}(g)$	The propagation time between UE $u \in \bar{N}_{ue}$ and gNB $g \in N_{gnb}$.

and transport network, and the processing time of the VNF instance. It is worth mentioning that each service is represented as a single VNF instance for the sake of simplicity. Although the problem formulation can be easily adapted to support more complex service function chains, it would dramatically increase the execution time of the proposed MILP-based algorithm without adding any significant value. Table 5.2 summarizes the notations used for the service requests.

Air Interface Capacity Calculation. The air interface capacity between gNB $g \in N_{gnb}$ and UE $u \in \bar{N}_{ue}$ is denoted by C^{u}_{g}, which is a function of Signal to Interference plus Noise Ratio (SINR) that can be computed through the following equation:

$$\forall g \in N_{gnb}, \forall u \in \bar{N}_{ue} : SINR_{g,u} = \frac{P^{g}_{tx} d^{-\delta}_{(g,u)}}{\mathcal{N}^2 + \sum_{k \neq g} P^{k}_{tx} d^{-\delta}_{(k,u)}} \tag{5.1}$$

where P^{g}_{tx} denotes the transmission power of gNB $g \in N_{gnb}$. It is worth noting that UEs will experience different signal strengths from the gNBs since cells are overlapping in the area of coverage. $d(g,u)$ is the euclidean distance between gNB $g \in N_{gnb}$ and UE $u \in \bar{N}_{ue}$, while δ represents the path loss coefficient and $\mathcal{N}$ is the noise power. Accordingly, if we define W as the system bandwidth, the maximum achievable air interface capacity C^{u}_{g} between gNB $g \in N_{gnb}$ and UE $u \in \bar{N}_{ue}$ can be computed as follows:

$$C^{u}_{g} = W \log(1 + SINR_{g,u}) \tag{5.2}$$

Based on the UE's Channel Quality Indicator (CQI) value, which can be obtained from the mapping table using the UE's SINR, we can compute the number of PRBs required to satisfy UE' data rate demand [107]. The CQI is determined in a way that corresponds to the highest Modulation and Coding Scheme (MCS), which also can be derived from the mapping table given in [107].

Given the throughput demand ω^{u}_{bwt} of UE $u \in \bar{N}_{ue}$, the number of required PRBs to meet the data request of the UE from gNB $g \in N_{gnb}$ can be computed as follows [65]:

$$\omega^{u,g}_{prb} = \frac{\omega^{u}_{bwt} T_{sbf}}{2 N_{sbc} N_{sym} N^{u,g}_{modb} N_{ant}} \tag{5.3}$$

where T_{sbf} is the duration of one sub-frame (1ms) and ω^{u}_{bwt} represents the throughput requested from the UE. N_{sbc} represents the number of sub-carries, which is equal to 12 sub-carries per PRB. N_{sym} represent the number of symbols per slot which is equal to 7 and we have 2 slots per sub-frame. Also, $N^{u,g}_{modb}$ and N_{ant}, respectively, represent the number of modulated bits per symbol for a given MCS and the number of antennas per gNB that is considered to be 2 in our scenario.

5.3.1 MILP-based Method

The described VNE problem has been formulated by employing MILP techniques. As mentioned earlier, three objectives are defined for the model, which are to minimize (i) service provisioning cost that is defined by equation (5.4), (ii) the transport network utilization given by equation (5.5), and (iii) the impact of VNF migration on users' quality of experience, defined by equation (5.6). Table 5.3 represents the variables used in the MILP model.

The objective (5.4) tends to minimize the service provisioning cost, which encompasses the cost of using computing, link transmission, and radio access network resources. While the costs of using link transmission and radio access network resources are the same for, respectively, all the links and gNBs, the cost of the computing resources depends on the type/location of the host node (e.g., edge, core, cloud). The closer the host node is located to the cloud, the more abundant and the

cheaper are its resources and, therefore, the cheaper is VNF instantiation.

$$Cost_M : min(\sum_{n\in N}\sum_{s\in N_{vnf}}\sum_{i\in N^s_{inst}} \xi^n_{cpu}\omega^i_{cpu}\chi^i_n + \sum_{u\in N_{ue}}\sum_{\bar{e}\in\bar{E}^u}\sum_{e\in E} \xi^e_{bwt}\omega^u_{bwt}\chi^{u,\bar{e}}_e + \sum_{u\in N_{ue}}\sum_{g\in N_{gnb}} \xi^g_{prb}\omega^{u,g}_{prb}\chi^u_g) \tag{5.4}$$

The following objective (5.5) aims at minimizing the bandwidth consumption of the transport network.

$$Link_M : min \sum_{u\in N_{ue}}\sum_{\bar{e}\in\bar{E}}\sum_{e\in E} \omega^u_{bwt}\chi^{u,\bar{e}}_e \tag{5.5}$$

This objective is particularly useful for the cases in which the transport network lacks of capacity, or the UE requested service is latency sensitive.

Finally, the goal of the last objective (5.6) is to minimize the impact of VNF migration on UEs' quality of experience. The need for this objective stems from the fact that the VNF migration may cause service interruption, which, in turn, may degrade the QoS experienced by the UEs. The effect of the VNF migration onto the UEs is minimized by intelligently selecting the VNF to be migrated. It is important to mention that this objective takes into account also the cost[1] of using CPU, transport network, and PRB resources like in Formula (5.4). As opposed to Formula (5.4), however, those resources have a very small coefficient in order to make sure that it does not affect the main argument defined in Formula (5.6).

$$Mig_M : min \sum_{n\in N}\sum_{s\in N_{vnf}}\sum_{i\in N^s_{inst}} N^{i\star}_n * I^i_n \tag{5.6}$$

In the following, we present the constraints that, regardless of the objective function, have to be satisfied for a solution to be valid.

Constraint (5.7) pertains to the UE association, making sure that each UE is connected to only one gNB, which has to have sufficient link capacity (enforced by formula (5.8)) and sufficient amount of PRBs in order to satisfy the UEs' data rate demand (enforced by constraint (5.9)).

$$\forall u \in \bar{N}_{ue} : \sum_{g\in N_{gnb}} \chi^u_g = 1 \tag{5.7}$$

$$\forall g \in N_{gnb} : \sum_{u\in\bar{N}_{ue}} \omega^u_{bwt}\chi^u_g < C^u_g \tag{5.8}$$

[1]Note that this is not shown in Formula (5.6) in order to avoid confusion.

$$\forall g \in N_{gnb} : \sum_{u \in \bar{N}_{ue}} \omega_{prb}^{u,g} \chi_g^u \leq \omega_{prb}^g \tag{5.9}$$

As stated before, our model assumes that each UE requests only one service. Thus, constraint (5.10) enforces each UE $u \in \bar{N}_{ue}$ to be connected to only a single service/VNF instance.

$$\forall u \in \bar{N}_{ue} : \sum_{n \in N} \sum_{s \in \bar{N}_{vnf}} \sum_{i \in N_{inst}^s} \chi_{u,n}^i = 1 \tag{5.10}$$

The following constraint guarantees that a VNF is spawned/instantiated only if at least one UE is mapped on that VNF.

$$\forall n \in N, \forall s \in N_{vnf}, \forall i \in N_{inst}^s : \sum_{u \in \bar{N}_{ue}} \chi_{u,n}^i - \mu * \chi_n^i \leq 0 \tag{5.11}$$

As mentioned earlier, the amount of computational resources on the MEC servers collocated with gNBs is limited and expensive, different from the cloud where we can find significantly more amount of resources at a much cheaper price. In other words, the more we get closer to the UE, the more resources get scarce and expensive. Therefore, before placing a service on a node, it should be checked if that node has a sufficient amount of resources to host the service, making sure that the number of CPU resources assigned to a VNF instances running on a node does not exceed the CPU capacity of that node (constraint (5.12)).

$$\forall n \in N : \sum_{s \in \bar{N}_{vnf}} \sum_{i \in N_{inst}^s} \omega_{cpu}^i \chi_n^i \leq C_{cpu}^n \tag{5.12}$$

Constraint (5.13) sets an upper bound on the number of UEs that can use the same VNF instance.

$$\forall n \in N, \forall s \in N_{vnf}, \forall i \in N_{inst}^s : \sum_{u \in \bar{N}_{ue}} \chi_{u,n}^i \leq C_{ue}^i(n) \tag{5.13}$$

Constraint (5.14) ensures that the virtual links can be mapped onto a substrate link as long as the link has sufficient capacity:

$$\forall e \in E : \sum_{u \in \bar{N}_{ue}} \sum_{\bar{e} \in \bar{E}} \omega_{bwt}^u \chi_e^{u,\bar{e}} < C_{bwt}^e \tag{5.14}$$

Constraint (5.15) indicates if the instance $i \in N^s_{inst}$ of the service $s \in N_{vnf}$ migarted from the node $n \in N$.

$$\forall n \in N, \forall s \in N_{vnf}, \forall i \in N^s_{inst} : \chi^{i\star}_n - \chi^i_n - I^i_n \leq 0 \tag{5.15}$$

The processing time $T^i_{proc}(n)$ of the i^{th} instance of service s on the node n is computed by constraint (5.16) considering the aggregated data to be processed by that service instance, while constraint (5.17) ensures that if the UE u uses that VNF instance ($\chi^i_{u,n} = 1$) then the VNF processing time $T^i_{proc}(u,n) = T^i_{proc}(n)$ is taken into account.

$$\forall n \in N, \forall s \in N_{vnf}, \forall i \in N^s_{inst} : \sum_{u \in \bar{N}_{ue}} \frac{\omega^u_{bwt}}{C^i_{proc}(n)} \chi^i_{u,n} - T^i_{proc}(n) = 0 \tag{5.16}$$

$$\begin{aligned} &\forall n \in N, \forall u \in \bar{N}_{ue}, \forall s \in N_{vnf}, \forall i \in N^s_{inst} : \\ &\mu \chi^i_{u,n} + T^i_{proc}(n) - T^i_{proc}(u,n) \leq \mu \end{aligned} \tag{5.17}$$

A similar approach is adopted by constraint (5.18) to compute the transmission time T^e_{tx} over the substrate link e, while constraint (5.19) handles the accurate transmission time computation over the virtual link $\bar{e}$.

$$\forall e \in E : \sum_{u \in \bar{N}_{ue}} \sum_{\bar{e} \in \bar{E}^u} \frac{\omega^u_{bwt}}{C^e_{bwt}} \chi^{u,\bar{e}}_e - T^e_{tx} = 0 \tag{5.18}$$

$$\forall e \in E, \forall \bar{e} \in \bar{E}, \forall u \in \bar{N}_{ue} : \mu * \chi^{u,\bar{e}}_e + T^e_{tx} - T^{u,\bar{e}}_{tx}(e) \leq \mu \tag{5.19}$$

Constraint (5.20) ensures that there is a continues path between the UE $u \in \bar{N}_{ue}$ and the instance $i \in N^s_{inst}$ of the service $s \in \bar{N}_{vnf}$.

$$\begin{aligned} &\forall m,n \in N, \forall \bar{e} \in \bar{E}, \forall u \in \bar{N}_{ue} : \\ &\sum_{e \in E^{n\rightarrow}} \chi^{e^{(n,m)}}_e - \sum_{e \in E^{\rightarrow n}} \chi^{e^{(n,m)}}_e = \begin{cases} -1 & \text{if } i = n \\ 1 & \text{if } i = m \\ 0 & \text{otherwise} \end{cases} \end{aligned} \tag{5.20}$$

where $E^{n\rightarrow}$ represents the links originating from node $n \in N$, while $E^{\rightarrow n}$ represents all the links entering node $n \in N$.

TABLE 5.3: Binary and continuous decision variables.

Variables	Description
χ_g^u	Indicates if UE $u \in \bar{N}_{ue}$ is associated to gNB $g \in N_{gnb}$.
χ_n^i	Indicates if instances $i \in N_{inst}^s$ of service $s \in N_{vnf}$ is running on node $n \in N$.
$\chi_n^{i\star}$	A parameter which shows the previous assignment of instance $i \in N_{inst}^s$ of service $s \in N_{vnf}$ on node $n \in N$.
$\chi_{u,n}^i$	Indicates if instances $i \in N_{inst}^s$ of service $s \in N_{vnf}$ is running on node $n \in N$ and assigned to UE $u \in \bar{N}_{ue}$.
$\chi_e^{u,\bar{e}}$	Indicates if the virtual link $\bar{e} \in \bar{E}$ belonging to the request by UE $u \in \bar{N}_{ue}$ is mapped on the substrate link $e \in E$.
I_n^i	If a migration has taken place for the instance $i \in N_{inst}^s$ of service $s \in N_{vnf}$ on node $n \in N$.
$T_{proc}^i(n)$	Processing time of instance $i \in N_{inst}^s$ of service $s \in N_{vnf}$ on node $n \in N$.
$T_{proc}^i(u,n)$	Processing time of instance $i \in N_{inst}^s$ of service $s \in N_{vnf}$ on node $n \in N$ for UE $u \in \bar{N}_{ue}$.
T_{tx}^e	Transmission time over link $e \in E$.
$T_{tx}^{u,\bar{e}}(e)$	Transmission time over link $e \in E$ for virtual link $\bar{e} \in \bar{E}$.

The delay of a service $s \in N_{vnf}$ is computed from the time the request is issued until the time the requested data is received by the UE. We consider the propagation delay, transmission delay, and the VNF computing delay for each UE $u \in \bar{N}_{ue}$. Both the air interface delay and the transport link delay are taken into account in the calculation of the propagation and transmission delay. Constraint (5.21) guarantees that the aggregated delay does not exceed the maximum delay budget defined for the UE u:

$$\begin{aligned} \forall u \in N_{ue} : & \sum_{n \in N} \sum_{s \in N_{vnf}} \sum_{i \in N_{inst}^s} T_{proc}^i(u,n) \\ & + \sum_{e \in E} T_{tx,prp}^{u,\bar{e}}(e) + \sum_{g \in N_{gnb}} T_{tx,prp}^u(g) \leq T_{max}^u \end{aligned} \tag{5.21}$$

5.3.2 Heuristic

Although the MILP model achieves the optimal solution in all the scenarios, it becomes computationally intractable with the increase in the network size. Therefore, to combat the scalability issue of the MILP model, this section presents a heuristic

Algorithm 2: Mig_H

Input: $(G, \bar{G})$
Output: UEs association, VNF placement and resource allocation;

Phase 1: Find the candidate gNBs;
for $u \in \bar{N}_{ue}$ **do**
  $cand\_gnb(u) \leftarrow \emptyset$;
  for $g \in N_{gnb}$ **do**
    $\omega_{prb}^{u,g} \leftarrow calcPRB(u,g)$;
    if $\omega_{prb}^{g} \geq \omega_{prb}^{u,g}$ ***and*** $C_g^u \geq \omega_{bwt}^u$ **then**
      $cand\_gnb(u) \leftarrow g$;

Phase 2: Find the highest priority gNB and computing server for each UE and then allocate the resources;
for $u \in \bar{N}_{ue}$ **do**
  for $i \in N_{inst}^{s(u)}$ **do**
    for $n \in N$ **do**
      $serverPriority[u,i,n] \leftarrow calcPriority(u,i,n)$;
  $flag \leftarrow False$;
  • Sort the $cand\_gnb(u)$ in ascending order according to the # of PRBs;
  for $g \in cand\_gnb(u)$ **do**
    for $i,n \in serverPriority[u,i,n] \downarrow$ **do**
      $T_{proc}^{i}(u,n) \leftarrow calcProcDelay(u,i,n)$;
      $T_{tx,prp}^{u,\bar{e}}(g,n) \leftarrow calcLinkDelay(u,\bar{e},g,n)$;
      $T_{tx,prp}^{u}(g) \leftarrow calcAirDelay(u,g)$;
      $T_{tot} \leftarrow T_{proc}^{i}(u,n) + T_{tx,prp}^{u,\bar{e}}(g,n) + T_{tx,prp}^{u}(g)$;
      if $T_{tot} \leq T_{max}^{u}$ **then**
        $flag \leftarrow True$;
        break;
    if $flag$ ***is*** $True$ **then**
      • Allocate path $P_{g,n}$;
      • Allocate and update network resources;
      break;

Phase 3: Resource usage optimization and migration control;
for $n \in N$ **do**
  for $i \in N_{inst}^{s(u)}$ **do**
    for $m \in N$ **do**
      if i is mapped on m and n **then**
        if $T_{tot} \leq T_{max}^{u}$ ***and*** $N_{ue}^{i,m} + N_{ue}^{i,n} \leq C_{ue}^{i}(n)$ **then**
          for $u \in maped(i \rightarrow n)$ **do**
            • Migrate UEs to the VNF on node m;
            • Allocate and update network resources;

algorithm, as shown in the algorithm (2), that aims at reaching a near-optimal solution for the problem in a considerably shorter time.

Similar to the Mig_M algorithm, the objective of the proposed heuristic algorithm is to minimize the number of users affected by VNF migration. The algorithm is divided into three phases. The first phase aims at finding the list of the candidate gNBs $cand\_gnb(u)$ for each UE u. A gNB g is considered to be a candidate for the UE u only if that gNB has the required amount of PRBs $\omega_{prb}^{u,g}$ computed by formula (5.3) and higher air interface capacity computed by formula (5.2) in order to support the data rate demand of the UE u. This phase of the algorithm is of order $O(mn)$, in which m is the number of UEs and n is the number of gNBs.

The second phase of the algorithm attempts to find the highest priority gNBs and computing server for each request and allocate enough resources to accommodate the UE. As the first step, a 3D matrix ($serverPriority[u,i,n]$) is used to store each computing server's priority for hosting the requested service. The matrix is populated by a function called $calcPriority(u,i,n)$ that gives a score to each combination of UE, VNF instance, and node. The logic behind the $calcPriority(u,i,n)$ function is to prioritize placing VNFs at the same node compared to the previous run and associate the UEs to the same VNFs as before unless the UE requirement cannot be fulfilled with the current allocation. The function computes the priority of embedding the requested service type with different instances on different nodes for each of the given UEs. Many parameters are involved in calculating the priority of embedding a VNF instance on a node for a specific UE. When a UE was assigned to a VNF instance on a specific node in the previous run, the same assignment will get the highest priority — if not, assigning the UE to an instance of the same VNF type embedded in the previous run, which did not serve the UE get the highest priority. Next, if in the current run a VNF is embedded, the aim will be to reuse the same VNF instance for the other UE in the same batch that asks for the same service type. The last priority is to embed the requested service type on a node with the highest resource capacity. It is worth noting that the number of CPU resources needed for VNF instantiation and the amount of bandwidth required on the links are considered in the priority calculation process for all the cases. The next step is to sort the candidate gNBs for each UE in an ascending order based on the number of PRBs required to associate the UE to the corresponding gNB. After that, for each candidate gNB, the algorithm loops over all the servers, starting from the one with the highest priority. The VNF processing delay on the node, transmission, and propagation delay over the transport link and air interface are computed in each run. If the overall delay of a placement solution is lower than the maximum delay tolerance of the UE, it

TABLE 5.4: Service requirements.

Service type	**E2E delay tolerance**	**Data rate requirement**
Autonomous cars	$\leq 50ms$	$\geq 1Mbps$
Video streaming	$\leq 400ms$	$\geq 5Mbps$
Smart homes	$\leq 300ms$	$\geq 3Mbps$
Robotic	$\leq 150ms$	$\geq 1Mbps$

will be considered as the best solution and break the loop to allocate the required resources to the request. This process is repeated until all the requests are embedded on the substrate network. As noted, finding a proper placement and allocation is the dominant procedure in the second phase; in this regards, the time complexity of this phase is of order $O(mnkp)$, k is the number of VNF instances, and p is the number of nodes.

During the embedding process, we might encounter a situation in which many VNFs are embedded in the very first runs but just a few UEs associated with them. It happens due to the fact that with the arrival of new batches, UEs cannot be accommodated on the previous VNF instances due to latency violation, and new VNF instances will be instantiated for the newly arrived requests. In this regard, the third phase of the algorithm tries to remove the old VNFs with few UEs attached to them and re-associate the UEs with the recently deployed VNF instances. This procedure takes place in order to avoid over-utilization of CPU resources. This procedure might trigger a VNF migration on the condition that the delay requirement of all the UEs being served from that VNF as well as the new UEs that are expected to be associated with that VNF instance is fulfilled, and the total number of these UEs $(N_{ue}^{i,m} + N_{ue}^{i,n})$ does not increase the maximum number of UEs $C_{ue}^{i}(n)$ allowed for this VNF instance. During the migration process, VNF instances with less number of UEs will be preferred for migration. As a consequence, this will result in the minimization of the number of UEs affected by the VNF migration. This will be followed by the allocation and update of the network resources. It is worth mentioning that, in order to ensure the correctness of the solutions, we pass all the solutions found by the heuristic trough the same constraints defined for the Mig_M formulation defined in Section 5.3.1. The time complexity of this phase is of order $O(mnkp)$, since it needs to go over all the UEs, gNBs, VNF instance, and computing nodes. Overall, the complexity of the algorithm is $O(c_1mn + c_2mnkp + c_3mnkp)$, where c is a constant and negligible. Therefore, the complexity of the algorithm is of the order $O(mnkp)$.

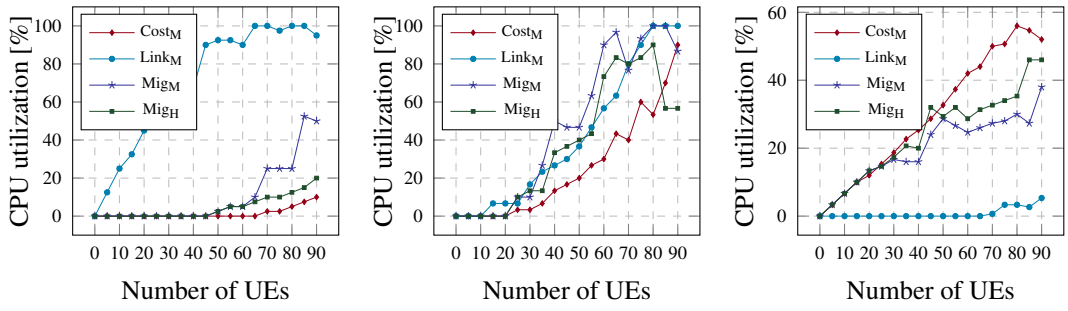
5.4 Performance Evaluation

The goal of this section is to compare the presented MILP-based and heuristic algorithms. We shall first describe the simulation setup used in our study. We will then discuss the outcomes of the numerical simulations carried out in Python using Gurobi mathematical optimization solver [15].

5.4.1 Simulation Environment

The mobile network considered in this work is composed of 6 nodes, out of which one is a cloud server, one is a core server, and the rest are gNBs, referred to as edge nodes. All of the edge nodes and the core node have a collocated MEC server. The cloud server is connected to the core server via a 2.5 Gbps backhaul link; whereas, the edge nodes are connected to the core server via 700 Mbps FH links. The edge nodes, the core, and the cloud have, respectively, 2, 6, and 30 CPU cores, each of which has a 3.4 GHz clock rate. The capacity of a VNF instance depends on the number of allocated CPU cores. We assume that at least a single CPU core is required in order to spawn/instantiate a VNF. The maximum number of UEs that can use the same VNF depends on the number of CPU cores allocated to that VNF, and it is assumed that a single CPU core can be used by 5 UEs at most. Thus, once a VNF is instantiated on a node, it can be used by a certain number of UEs under the condition of not violating the E2E latency of the UEs connected to the VNF instance.

Every minute, which is considered a single time slot, a new batch arrives composed of 5 UEs, each of which is making a service request. For the sake of simplicity, each service type is treated as a single VNF. Upon receiving the service requests, the algorithms try to associate the UEs to the gNBs, place the VNFs on the computing servers, and allocate enough resources to the spawned VNF. We consider 18 batches of service requests (90 UEs in total) due to the scalability issue of the MILP-based algorithms. We assume that 8 types of VNFs exist, each of which belongs to one of the three service classes identified by their data rate and E2E delay tolerance (strict, medium, and loose) requirement. Specifically, the data rate and the E2E delay tolerance are selected, respectively, from the range of $[T_{max} <= 50; 51 <= T_{max} <= 200; 201 <= T_{max} <= 500]$ ms and $[1-2; 3-5; 6-12]$ Mbps for the aforementioned service classes. Examples of the services together with their E2E delay tolerance and data rate requirements are given in Table 5.4. If the UE association and his service request is accepted, the service provider has to guarantee that the required data rate and the E2E delay tolerance are always satisfied.



(A) CPU utlization at the edge. (B) CPU utilization of core. (C) CPU utilization of cloud.

FIGURE 5.2: CPU utilization of edge, core and cloud nodes.

For the sake of simplicity, for both downlink and uplink, the data size and data rate are considered to be the same. Although the Transmission Time Interval (TTI) can be dynamically tuned in 5G networks, we consider $T_{tx} = 1ms$ as fixed TTI. The transmission time and processing time for each UE are computed considering all other UEs mapped, respectively, on the same fronthaul/backhaul link and VNF. Specifically, T_{tx} for the UEs using the same fronthaul or backhaul link at the considered moment is obtained by dividing the aggregated data size by the respective link rate. As for the processing time T^i_{proc} of a service/VNF, it is obtained by dividing the aggregated data demand on the VNF by the processing capacity of that VNF, which is the product of the number of CPU cores allocated to that VNF instance, clock rate of each CPU edge nodes the number of CPU cycles required to process one bit of information.

5.4.2 Simulation Results

The reported results are the average of 5 simulations with 95% confidence intervals. During each simulation, the algorithms try to sequentially associate to the network and embed the service requests of up to 90 UEs, whose requests arrive in batches each composed of 5 UEs. It is important to mention that all the algorithms employ a dynamic embedding strategy, that is, with the arrival of a new service request, the request along with the ones that have been previously embedded are re-embedded. Thus, with every embedding, the optimal embedding solution is found for all the UEs' requests.

CPU Utilization. As previously mentioned, a single CPU core can be used by a maximum of five UEs. Consequently, the number of UEs that can use the same VNF instance depends on its capacity in terms of the number of CPUs. Therefore, the CPU utilization of a node is the ratio between the number of UEs using the VNFs

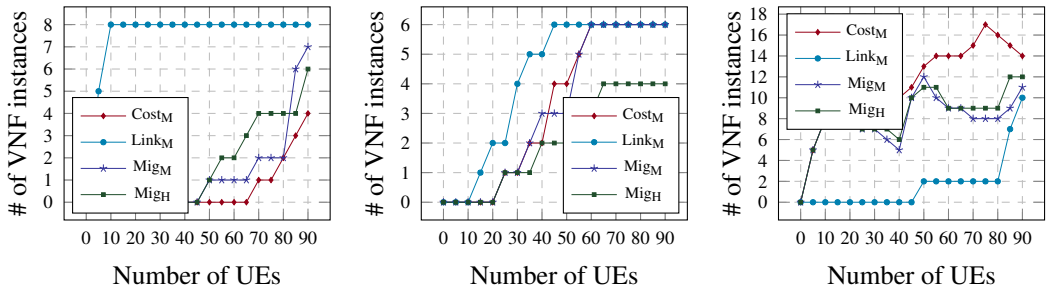
deployed on that node and the total number of UEs that can use this node, which is the multiplication of the number of the CPU cores of the node and the number of UEs that can use the same CPU core.

Figure 5.2 shows the CPU utilization on all the computing nodes for a single simulation run. Figure 5.2a depicts CPU utilization of the edges as a function of the number of UEs for all the algorithms. As can be inferred, the $Link_M$ algorithm begins the process of VNF placement by utilizing edge resources. This stems from the fact that the $Link_M$ algorithm aims at minimizing the transport network utilization, which is achieved by embedding the service requests at the edge servers, as close to the UEs as possible. Due to the scarcity of the processing resources at the edge, however, $Link_M$ shortly runs out of the edge resources and starts utilizing the core resources, as shown in Fig. 5.2b. For what concerns the cloud resources (see Fig. 5.2c), we can observe that $Link_M$ starts embedding VNFs in the cloud when 75 UEs are making a service request, achieving the lowest CPU utilization.

A reverse trend can be observed for the $Cost_M$ objective in terms of CPU utilization at the computing nodes. Specifically, it can be observed that $Cost_M$ tends to instantiate the VNFs starting from the core. This is due to significantly more processing resource availability at the cloud, compared to the edge and the core, which makes the total embedding cost much cheaper, regardless of the extra transport resource consumption. As expected, for the same reason, the CPU utilization at both edge and core is the smallest in most cases compared to the ones achieved by the rest of the algorithms.

As for the MILP-based and heuristic algorithms (i.e., Mig_M, Mig_H), their CPU utilization at the edge and the cloud lies somewhere in between the ones achieved by $Cost_M$ and $Link_M$, while it somewhat resembles to the CPU utilization of these algorithms at the core. More specifically, we can observe that up to 45 UEs, both Mig_M and Mig_H perform similar to $Cost_M$ since also they start instantiating VNFs from the cloud. When the number of UEs increases, however, it increases the transmission time over the FH/BH links (T_{tx}), which, in turn, results in the forthcoming requests being embedded on the core and the edges in order to avoid triggering VNF migrations. This is justified by the fact that the migration objectives strive to minimize the impact of VNF migration on UEs' QoE.

Number of VNFs. VNF instantiation requires computing resources, which incurs higher management costs on the network. In order to get an insight into how are the VNFs distributed across the computing nodes, let us analyze Fig. 5.3, which shows the result of a single simulation run. We can observe that after the second



(A) Number of VNFs at edge. (B) Number of VNFs at core. (C) Number of VNFs at cloud.

FIGURE 5.3: Number of VNF instance at edge, core and cloud nodes.

batch embedding (10 UEs), $Link_M$ runs out of computing resources at the edges, employing all of their CPU cores. Although this means that the edges can no longer host new VNF instances, it does not restrict the UEs to use the VNF instances already available on the edges and, therefore, increase their utilization, as shown in Fig. 5.2a. Similarly, $Link_M$ saturates also the CPU cores of the core node by instantiating 6 VNFs when there are 45 UEs making network association and service request; while, as expected, it utilizes a small portion of the cloud node by ultimately instantiating 10 VNFs (see Fig. 5.3c).

Regarding the $Cost_M$ algorithm, it is interesting to note that, even though it achieves the least amount of CPU utilization at the core node up to 80 UEs (see Fig. 5.2b), it instantiates more VNFs at the core up to 55 UEs compared to both of the migration algorithms (Mig_M, Mig_H), as displayed in Fig. 5.3b. In essence, this means that the VNFs instantiated by Mig_M and Mig_H on the core node, although fewer, are utilized by more UEs. For what concerns to the number of VNFs instantiated by $Cost_M$ on the edges and cloud, plotted, respectively, in Fig. 5.3a and Fig. 5.3c, they follow the same pattern of the CPU utilization at their corresponding nodes.

As for the MILP-based and heuristic migration algorithms, their performances resemble each other, especially at the cloud. In general, it can be observed that Mig_M and Mig_H utilize the VNF instances more efficiently compared to the rest of the algorithms since with the same number of VNFs, they achieve a higher CPU utilization in most of the cases. This is a consequence of the fact that Mig_M and Mig_H strives to minimize the number of VNF migrations and their effect onto the UEs' QoS, leading to their higher utilization. As for the fluctuating behavior of the algorithms in Fig. 5.2 and Fig. 5.3a, it is due to the VNF migrations, which will be analyzed in Fig. 5.5.

As stated before, the more CPU cores are assigned to a VNF instance, the more is its processing capacity, resulting in faster execution of UEs' tasks; nonetheless, the much more is also its instantiation cost. While Fig. 5.3 shows the total number of VNF instances across the edges, the core, and the cloud, it does not show the

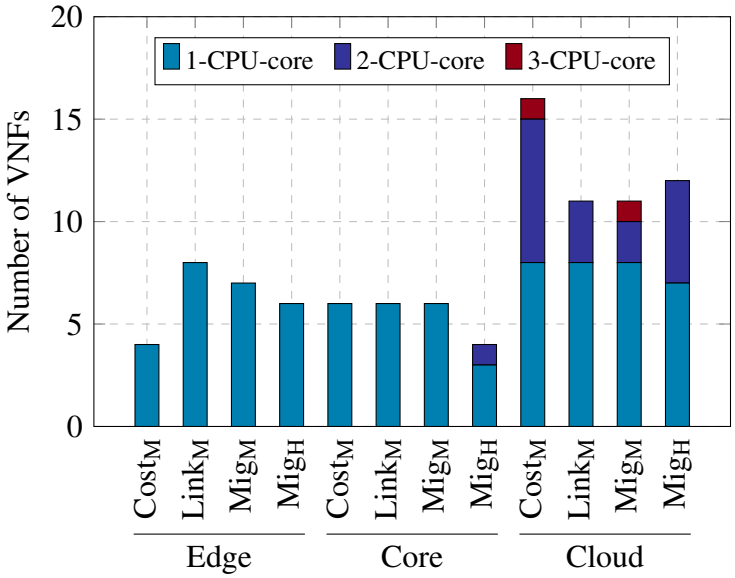


FIGURE 5.4: Number of VNF instances with different capacity at edge, core, and cloud.

capacity of those VNFs. In order to have a better understanding of how the CPU cores of the computing nodes are allocated to the VNF instances, and how many VNFs with different capacities are instantiated on the edges, the core, and the cloud, let us analyze Fig. 5.4. It can be observed that after all embeddings only 1-CPU-core VNFs are instantiated on the edge nodes. This is due to the fact that the computational capacity of the edge nodes is very limited in comparison with the core and cloud nodes. Therefore, the algorithms prefer to instantiate more VNF types on the edges in order to meet the E2E latency requirement of the UEs with various service/VNF request rather than to instantiate a few of them with more computational capacity. For the same reason, similar behavior can be observed for the core (only Mig_H instantiates 2-CPU-core VNFs), which has significantly less computing capacity compared to the cloud. Moreover, the core is far closer to the UEs since it requires no BH resources, curtailing the E2E latency experienced by the UEs. As for the cloud node, we can observe that all the algorithms instantiate 2-CPU-core VNFs, while Mig_M and Mig_H instantiate also 3-CPU-core VNFs. The rationale behind this behavior is that the cloud node has plenty of CPU cores, which make the VNF instantiation much cheaper. As expected, among all the algorithms, the highest number of VNFs with different capacities is instantiated by $Cost_M$ since, as opposed to the rest of the algorithms, it always prefers to embed the VNFs at the cloud as long as all of its constraints are satisfied.

Number of VNF migrations and their effect on users. The migration of VNFs can have a severe effect on the overall performance of the network and the UEs. For instance, a VNF migration may degrade its UEs' QoS possibly resulting in

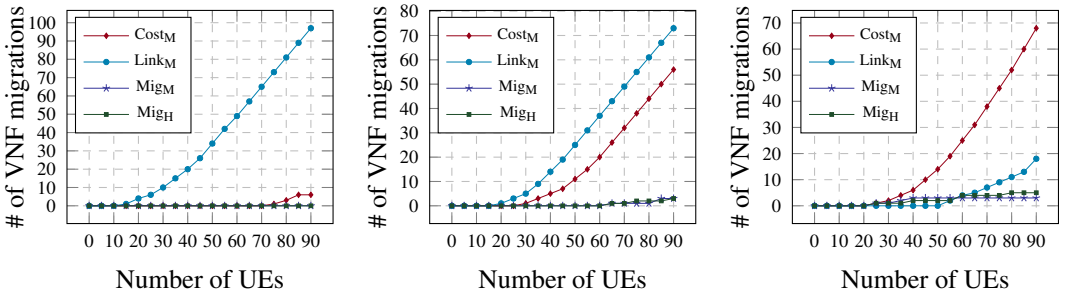
their service interruption. While Fig. 5.3 shows how many VNFs are embedded at the computing nodes, it does not show the VNFs that migrate across the computing nodes. In order to see when and from which computing node the VNF migrations take place, let us analyze the cumulative number of VNF migrations taken from a single simulation run displayed in Fig. 5.5.

As shown in Fig. 5.5a, VNF migration from the edge starts shortly after embedding a few batches of requests by the $Link_M$ algorithm that, regardless of the requested service type, tries to embed the VNFs at the edge to minimize the transport network utilization. When the edges are no longer able to accommodate more VNFs due to scarce CPU cores, it starts migrating those VNFs from the edge that do not have strict E2E latency requirements. From the edge node, these VNF migrations take place either to host a newly arriving service request that has a strict latency requirement, or to increase the capacity of the migrated VNF by allocating more CPU cores aiming at serving more UEs and satisfy their E2E latency requirements. For the same reason, $Link_M$ is the first algorithm that starts migrating VNFs from the core (see Fig. 5.5b), while, as expected, the VNF migrations from the cloud for $Link_M$ is triggered much later (see Fig. 5.5c).

An opposite behavior is observed from $Cost_M$ across all the computing nodes, which due to its objective function, starts VNF instantiation from the cloud. Once the number of UEs that are served from the VNFs instantiated in the cloud increases, it increases the utilization of the FH and BH links, which in turn, increases the transmission time over those links. As a consequence, with the arrival of more UEs' service requests, this may result in the E2E latency constraint violation especially for those UEs that have a strict latency demand, unless some of the VNFs are migrated from the cloud. As expected, only a few VNF migrations are induced from the edge for $Cost_M$.

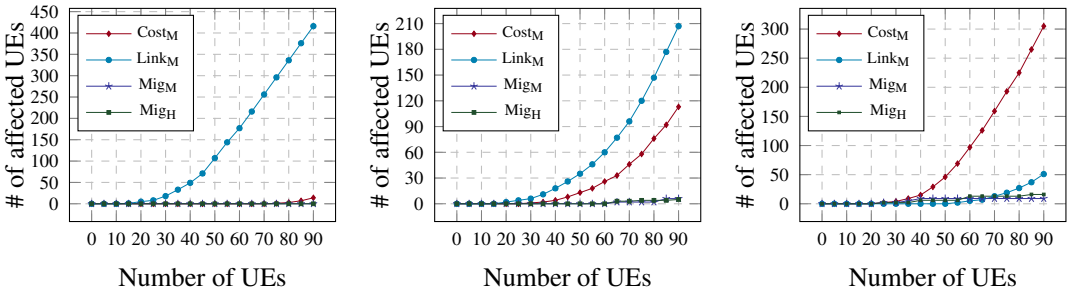
For what concerns the MILP-based and heuristic migration algorithms, thanks to the fact that they aim at minimizing the number of VNF migrations and their effect onto the UEs QoS, they exhibit similar performance. Specifically, they trigger no VNF migration from the edge and only a few VNF migrations from the core and the cloud.

As already mentioned, each VNF migration may affect the UEs that are using that VNF. In order to get an insight into how many UEs are affected due to the VNF migrations, let us analyze Fig. 5.6. It can be observed that the number of UEs affected by the VNF migrations closely follows the same pattern of the VNF migrations at all the computing nodes (edges, core, and cloud) shown in Fig. 5.5. This is because



(A) Number of VNF migrations from edge. (B) Number of VNF migrations from core. (C) Number of VNF migrations from cloud.

FIGURE 5.5: Cumulative number of VNF migration from edge, core, and cloud.

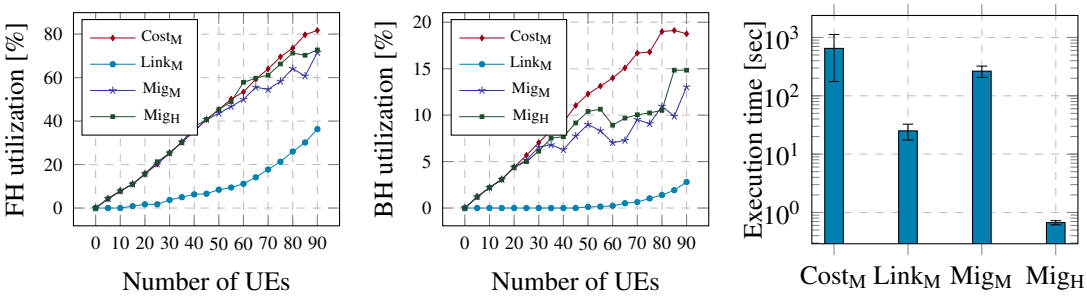


(A) No. of affected UEs at edge. (B) No. of affected UEs at core. (C) No. of affected UEs at cloud.

FIGURE 5.6: Cumulative number of UEs affected by VNF migration from edge, core, and cloud.

there is a direct relationship between the number of VNF migrations and the number of UEs affected by those VNF migrations. The more are the migrated VNFs, the more are the UEs affected by those VNF migrations. Thus, $Cost_M$, $Link_M$, and Mig_M along with $Link_H$ affect, respectively, the most, less and the least number of UEs.

Link utilization. Figure 5.7a and Fig. 5.7b illustrate, respectively, the FH and BH link utilization as a function of the number of UEs for a single simulation run. We can observe that $Cost_M$ and $Link_M$ achieve, respectively, the highest and the lowest FH and BH link utilization. This is justified by the fact that $Cost_M$ tends to utilize



(A) Fronthaul link utilization. (B) Backhaul link utilization. (C) Execution time.

FIGURE 5.7: FH and BH Link utilization in the entire network and execution time.

the cloud node resources as long as it does not violate the E2E latency constraints imposed by the service requests, therefore consuming also FH and BH link resources. Conversely, $Link_M$ aims at minimizing the transport network consumption in the network, therefore achieving the lowest FH and BH link utilization. For what concerns the migration algorithms, they experience FH and BH utilization which is close to the one's of $Cost_M$ algorithm when there are around 50 service requests; whereas, it resembles more to $Link_M$ when the number of requests increases.

Execution time. The main intention of the proposed heuristic algorithm is to combat the scalability issue of the MILP-based algorithms, which become computationally intractable when large substrate networks and more complex service requests composed of multiple VNFs are considered. The results given in Fig. 5.7c demonstrate the substantial improvement of the heuristic algorithm compared to its MILP-based counterparts in terms of execution time. Although the heuristic, due to its sub-optimal mapping solutions, performs poorer in terms of CPU utilization, the number of VNF migrations, and the number of affected UEs by migration, it proves to be competitive and also applicable to extensive size networks in real-world scenarios.

Figure 5.7c depicts the execution time of all the algorithms. It is obvious that the $Cost_M$ has much longer execution time compared to the other algorithms, which is due to the more parameters involved in the objective function. Moreover, the execution time of the Mig_M is higher than the $Link_M$. On the other hand, the execution time of the heuristic algorithm is much smaller, and it can reach a near-optimal solution in a matter of seconds.

5.5 Discussion

This chapter compared three strategies for solving a joint user association, SFC placement, and resource allocation problem in MEC-enabled 5G networks. Based on the reported results, we can conclude that $Link_M$, although saved the transport network resources and, therefore, is suitable to be used in the network segment that lacks the transport network capacity, and for the services that have a stringent latency requirement, triggered the highest number of VNF migrations from the edge nodes and the core node, thereby affecting the QoS for most of the UEs. $Cost_M$ instead is more appropriate to be used for the services that are latency tolerant and in the parts of the network that have sufficiently high transport network capacity since it aims at minimizing the service provisioning cost by instantiating VNF starting from the

cloud node. Nonetheless, like $Link_M$, $Cost_M$ yielded a significantly high number of VNF migrations especially from the cloud node, degrading the QoS for many UEs. As for Mig_M, it demonstrated to achieve better performance across all evaluation metrics compared to $Link_M$ and $Cost_M$ algorithms, which are two extremes in terms of employing the edge resources and the transport network resources. Thus, Mig_M found a better trade-off between the computational capacity of the computing nodes and the FH/BH bandwidth, resulting in a negligible number of VNF migrations. Finally, at the expense of suboptimal UE association and SFC placement compared to its MILP-based counterpart, Mig_H demonstrated the fastest execution time, making it suitable for larger-scale problems.

Chapter 6

User Association and SFC Lifecycle Management

This chapter studies the problem of a joint user association, service functions chain placement, and VNF scaling with a particular emphasis on analyzing the trade-offs between the VNF scaling strategies. Specifically, we compare vertical, horizontal, and hybrid VNF scaling strategies by formulating an ILP problem that minimizes the service provisioning cost for the MNOs, while satisfying users' data rate requirements. The users' service requests are represented as SFCs composed of the end-to-end mobile network components (e.g., gNBs, 5G core network VNFs, and application VNFs). Finally, we devise a heuristic algorithm to tackle the scalability issue of the ILP-based approach.

6.1 Overview

The rapid change in the mobile data traffic demand calls for efficient approaches to dynamically adjust the mobile network's capacity according to the demand. MNOs shall increase/decrease the capacity of both the 5G core network and application VNFs upon the need, ensuring optimal resource utilization and lowering the service provisioning cost. This is where the vertical, horizontal, and hybrid VNF scaling strategies come into play. While the vertical VNF scaling implies that the existing VNF is resized upon the need, adding/removing computational, memory, or storage resources, in the case of the horizontal VNF scaling, another instance of the same VNF is spawned/terminated. Although horizontal scaling ensures high scalability and reliability of the service, it suffers from increased resource consumption and state

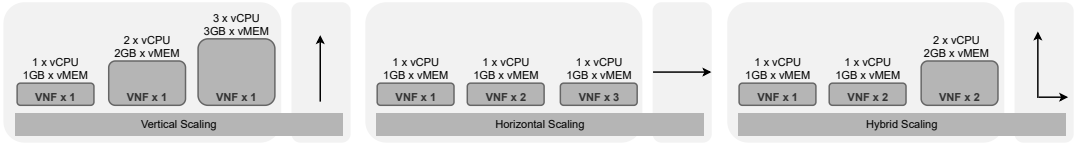


FIGURE 6.1: Horizontal, vertical, and hybrid VNF scaling.

migration challenges. On the other hand, while vertical scaling provides higher utilization of resources, thereby creating resource-optimized VNFs, its lower scalability and inability to change the VNF host significantly affect its practical implementation. Since both scaling strategies have their pros and cons, applying only vertical or horizontal scaling strategy cannot perform well in all the scenarios. This is why it is important to consider the so-called hybrid VNF scaling strategy, in which it is possible to perform either vertical or horizontal VNF scaling depending upon the need. However, it is a non-trivial task to decide which type of scaling to perform for a specific VNF since there is a number of parameters (e.g., the VNF type, its resource requirements) to take into account.

After performing VNF scaling, the placement of the VNF is another challenge that requires careful considerations. On the one hand, the interconnections between VNFs composing an SFC must be taken into account in order to make an optimal placement decision. On the other hand, the resource scarcity of the MEC servers at the network edges (e.g., collocated with gNBs) must be considered in order to efficiently utilize the network resources while at the same time satisfying the QoS requirement of the requested applications/services.

6.2 VNF Scaling Strategies

Since the main focus of our paper is on the VNF scaling problem, we shall first introduce the VNF scaling strategies studied in our work. The topmost figure of Fig. 6.1 illustrates the horizontal VNF scaling strategy. It is assumed that an autoscaling group is defined with a minimum and maximum number of VNF instances, and that there is a VNF template (i.e., VNF descriptor) with specific resource requirements that is used to spawn VNF instances. In case of a scale-out operation, one or multiple instances of the same VNF are deployed with identical resource flavor, using the predefined VNF template, while in the case of a scale in operation, one or more of the available VNF instances are terminated.

In the case of a horizontal VNF scaling, it is also assumed that the new VNF instance is preferably deployed on a different node, thus, guaranteeing high availability of the VNF/service, which is one of the prominent advantages of this VNF scaling strategy. Another advantage of the horizontal VNF scaling is its scalability, which is achieved due to small resource footprint of the VNFs.

The middle figure in Fig. 6.1 displays the vertical VNF scaling strategy. This type of VNF scaling implies that there is no change in the number of VNFs, while the current VNF is resized adding/curtailing certain resources (e.g, CPU, MEM, or both) in case of VNF scale up/down operation. In order to guarantee service continuity of the VNF, it is assumed that the VNF is terminated only after deploying its resized instance. The downside of the approach, however, is that it requires availability of resources on the host node even during the scale-down operation of the VNF, as opposes to the scale-in operation in the horizontal VNF scaling strategy. The vaunted benefits of the vertical VNF scaling include enabling the creation of a CPU-optimized or MEM-optimized VNF; that is, if only more CPU is required for a VNF, for example, then a new (i.e., resized) VNF instance could be spawned with more CPU resources, while leaving the MEM resource the same. This is in contrast to the horizontal VNF scaling strategy, which would just instantiate another VNF allocating both CPU and MEM resources even without the need for an extra MEM resource. While the vertical VNF scaling is significantly less scalable, it is a better strategy in terms of data consistency than its horizontal counterpart. This is because the resized VNF instances are preferably placed on the same host node exempting the need for the VNF/application state transfer from one node to another in case the considered VNF is stateful. For more details on the characteristics of these VNF scaling strategies, we refer the reader to [108].

As mentioned above, both the vertical and horizontal VNF scaling strategies have pros and cons. Each of these strategies perform better in a specific scenario. Therefore, more benefits can be reaped by using the hybrid VNF scaling strategy shown on the right-hand side of Fig. 6.1. This strategy incorporates both vertical and horizontal VNF scaling approaches, thereby enabling selection of the most appropriate approach for a VNF while considering a number of parameters such as the VNF type, its resource utilization and the QoS requirements, etc.

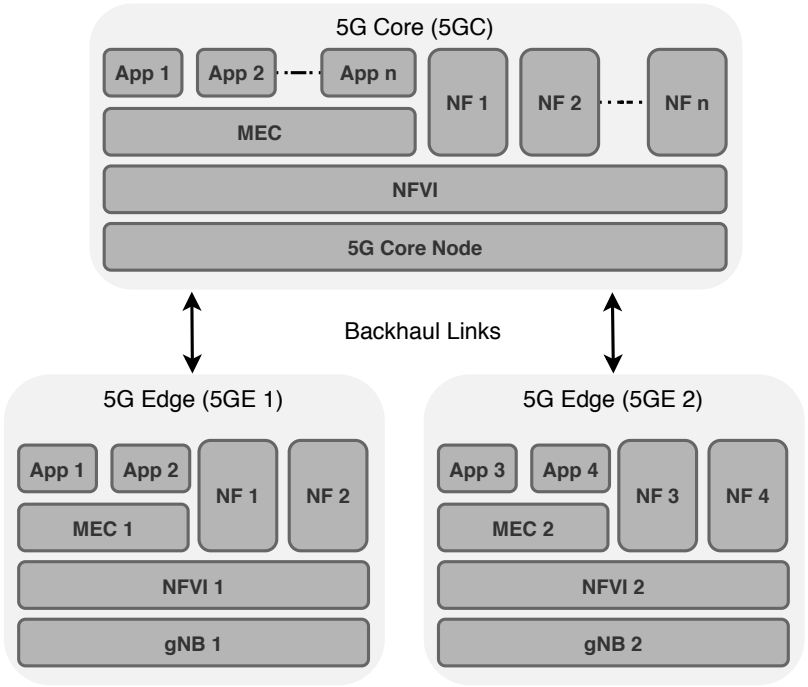
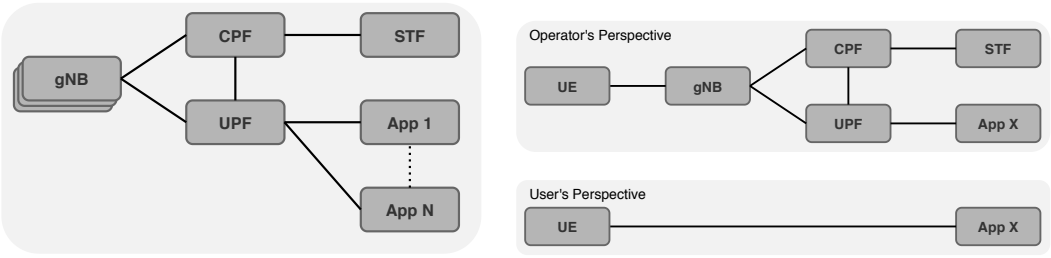


FIGURE 6.2: Physical architecture of the mobile network.

6.3 Problem Statement

The physical architecture of the mobile network considered in this study is composed of a 5G Core (5GC) and 5G Edges (5GEs), with the latter encompassing a gNB, as shown in Fig. 6.2. The 5GC is connected to the 5GEs by means of backhaul links. While both the 5GC and the 5GEs have deployed NFVI, the one of the 5GC possesses significantly more computational (vCPU) and Memory (vMEM) resources compared to that of 5GEs. By virtualizing the underlying hardware resources, NFVIs are capable of hosting VNFs **etsi2013gs**. It is assumed that each NFVI has already hosted MEC server as a VNF, which in turn is capable of hosting UE requested application VNFs [109]. Apart from the MEC server, the NFVIs can also host 5GC NFs such as UPF and Control Plane Function (CPF) as detailed in the subsequent section, which are also deployed as VNFs.

The mobile network's logical architecture is composed of gNBs, 5GC VNFs, and various application VNFs, as depicted in Fig. 6.3a. It is worth to mention that, thanks to NFVI and MEC servers, the logical mobile network can be mapped to either only a 5GE node or a composition of 5GE and 5GC nodes in the physical mobile network architecture depicted in Fig. 6.2. The 5GC VNFs are grouped into Stateful Functions (STFs), CPFs, and a UPF. While for the sake of simplicity, it is assumed that STFs and CPFs can be deployed as single VNFs, they consist of multiple stateful and control plane functions, respectively. For instance, the STFs in



(A) Logical network architecture. (B) UE request.

FIGURE 6.3: Logical mobile network architecture and UE request.

the SBA are composed of Network Repository Function (NRF), Unstructured Data Storage Function (USDF), and Unified Data Repository (UDR), while the CPFs contain network functions such as access and AMF, and SMF. STFs store UEs' subscription data, dynamic state data, application data, as well as the profiles of different NF instances. CPFs instead handle all the control plane signaling between the NFs and the UEs, performing authentication, session, and mobility management for the UEs. The UPF, on the other hand, is in charge of routing and forwarding the packets received either by the UEs through the gNBs in the uplink direction or by the applications deployed on the MEC servers in the downlink direction. Note that there may be multiple applications running on the MEC server, which can be accessed through the UPF as per ETSI [104]. For more information on 5G SBA and the functionalities of its NFs, we refer the reader to [110].

In Fig. 6.4 we show the message sequence diagram that illustrates a simplified call setup procedure and interactions between different SFC components (i.e., gNBs, CPFs, STFs, UPFs, and APPs). The procedure is initiated by a single UE, with a request being transmitted over a wireless channel to the gNB. Firstly, the user authentication starts with a gNB generating a request for a CPF, which further cooperates with an STF in order to store and handle user subscription data. Secondly, the gNogNB sends a session request to the CPF, which selects a corresponding UPF VNF. As UPF plays a pivotal role in data transfer, it is then used to interconnect the gNB and the application, thereby transferring the data between these two entities.

It is possible to deploy various applications on the mobile network in order to serve the requests of the UEs. While from the UE perspective, the application request is quite simple, it is more complex from the MNOs' perspective, as Fig. 6.3b illustrates in the upper and the lower parts, respectively. This is because in order to set up a network communication for a UE, regardless of the kind of the procedure the UE performs (e.g., attachment, handover, etc.), the UE has to be associated with a gNB, which, in turn, should establish a data plane and a control plane communication with

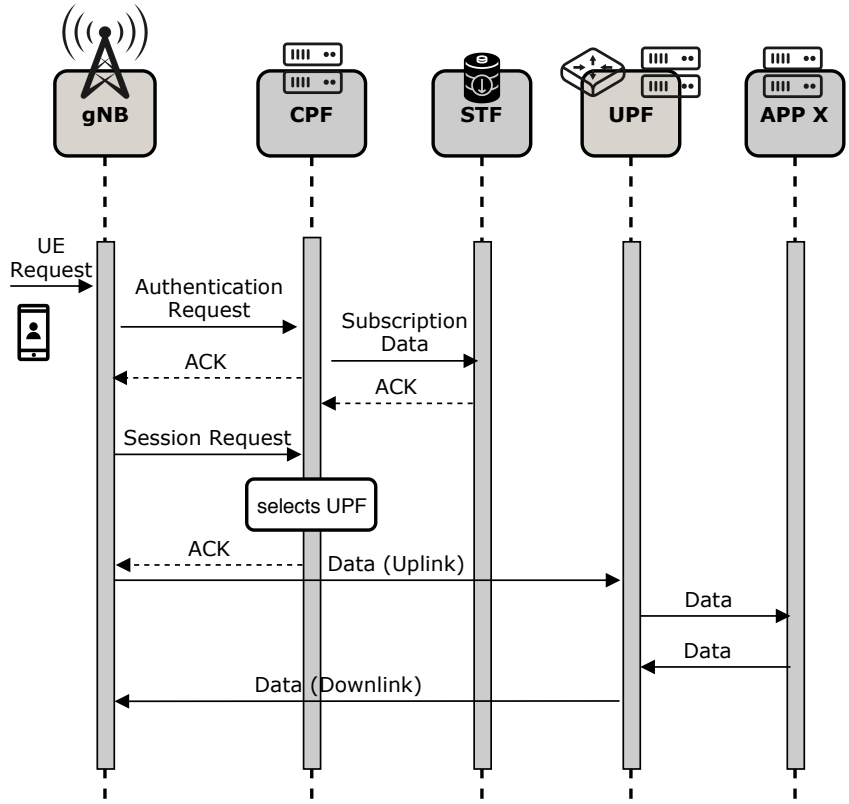


FIGURE 6.4: Sequence diagram of the UE request (to update it with regards to different types of UEs (i.e., data, voice)).

their respective UPF and CPFs, and access UE-related subscription/state information from STFs. Hence, the requests of the UEs are represented as SFCs encompassing the components of the end-to-end mobile network, as illustrated in Fig. 6.3b.

The optimal placement of these SFCs depends on various factors, including the resource availability, their cost, and QoS requirements of the requested service, such as the date rate and the E2E latency. While the E2E latency demand of an SFC plays a major role in the placement decision of the SFC, it is not considered in this work since the main focus of this study is on the VNF scaling strategy. For a detailed consideration of the SFC placement with an E2E latency requirement, we refer the reader to our previous study [111]. The problem of joint UEs' association, SFC placement, and VNF scaling can be stated as follows:

Given: a 5G mobile network with i) each node (e.g., 5GEs, 5GC) having a certain amount of vCPU and vMEM resources, ii) the transport network with the capacity of the backhaul links, and iii) a number of UEs making application requests with specific data rate demand.

Find: association of the UEs, the placement of the application SFCs, as well as the appropriate VNF scaling, if needed.

TABLE 6.1: Mobile network parameters.

Parameter	Description
G_{net}	Mobile network graph.
N_{net}	Set of nodes/DCs in G_{net}.
N_{5gc}	The 5GC in G_{net}.
N_{5ge}	Set of 5GEs in G_{net}.
N_{gnb}	Set of gNBs, one per 5GE in G_{net}.
$N_{upf,cfp}$	Set of user plane and control plane function VNFs.
$N_{stf,app}$	Set of stateful function and application VNFs.
N_{vnf}	Set of all VNFs, $N_{vnf} = N_{upf} \cup N_{cpf} \cup N_{stf} \cup N_{app}$.
N^v_{flv}	Flavours of the VNF v, $N^v_{flv} = N^v_{v-flv} \cup N^v_{h-flv}$.
E_{net}	Set of backhaul links in G_{net}.
$\omega_{prb}(g)$	PRB resources of gNB g n.
$\omega_{c,m}(n)$	Computational and memory resources of the node n.
$\omega^{vnf}_{c,m}(f)$	Computational and memory resources of the flavour $f \in N^v_{flv}$ of the VNF $v \in N_{vnf}$.
$\omega^{upf,app}_{t}(f)$	Throughput of flavour $f \in N^{upf,app}_{flv}$ of UPF and application VNF.
$\omega^{cpf}_{e}(f)$	Number of events supported by CPF flavour $f \in N^{cpf}_{flv}$.
$\omega^{stf}_{q}(f)$	Number of queries supported by STF flavour $f \in N^{stf}_{flv}$.
$\omega_b(e^{nm})$	Capacity of the link $e^{nm} \in E_{net}$.
$loc(n)$	Geographical location of the node $n \in N_{net}$.
$\delta(g)$	Coverage radius of the gNB $g \in N_{gnb}$ (in meters).

Objective: minimize the MNO's service provisioning cost.

Note that the mobile network/infrastructure provider is assumed to be the same entity providing the applications/services implemented by the SFCs. The proposed optimization approach, however, can be easily adapted to consider also the case in which these entities are different.

6.4 Proposed Methods

Mobile Network Model. Let $G_{net} = (N_{net}, E_{net})$ be an *undirected* graph modelling the physical architecture of the mobile network, where $N_{net} = N_{5ge} \cup N_{5gc}$ is the union of the set of 5GEs and a 5GC. There is a one-to-one mapping between a 5GE and a gNB, and it is assumed that the gNBs have sufficient amount of PRBs in order

TABLE 6.2: UE request parameters.

Parameter	Description
G_{req}	UE request graph.
N_{ue}	Set of UEs in G_{req}.
N^{ue}_{vnf}	Set of VNFs in G_{req}.
$N^{ue}_{app}(u)$	Application VNF requested by UE $u \in N_{ue}$.
$\omega^{ue}_{d}(u)$	Data rate requested by UE $u \in N_{ue}$.
$\omega^{ue}_{prb}(u)$	Number of PRBs needed to satisfy $\omega^{ue}_{d}(u)$.
$\omega^{ue}_{e}(u)$	Number of events generated by UE $u \in N_{ue}$.
$\omega^{ue}_{q}(u)$	Number of queries generated by UE $u \in N_{ue}$.
$E_{req}, E_{req}(u)$	Set of all virtual links, and that of the UE $u \in N_{ue}$.

to meet the data rate demand of the services requested by the UEs. Each network node $n \in N_{net}$ has a certain amount of vCPU $\omega_c(n)$ and vMEM $\omega_m(n)$ resource. Additionally, each node $n \in N_{net}$ is associated with a geographic location $loc(n)$, as x, y coordinates while each gNB $g \in N_{gnb}$ is also associated with a coverage radius of $\delta(g)$, in meters. E_{net} is the set of backhaul links interconnecting the 5GEs with the 5GC. An edge $e^{nm} \in E_{net}$ exists if and only if there is a connection between $n, m \in N_{net}$. A weight $\omega_b(e^{nm})$ is assigned to each edge $e^{nm} \in E_{net} : \omega_b(e^{nm}) \in \mathbb{N}^+$ representing its capacity, in Gbps.

All the nodes (i.e., 5GEs and 5GC) are able to host both 5GC network VNFs as well as application VNFs on their MEC server, which, in turn, is deployed as a VNF, however, is not a subject of scaling in our study. Each VNF $v \in N_{vnf}$ may have multiple flavours $N^v_{flv} = N^v_{v-flv} \cup N^v_{h-flv}$, each representing either another instance of the same VNF with identical amount of resources, referred to as horizontal flavour $f \in N^v_{h-flv}$, or a resized version of that VNF with more/less vCPU, and/or vMEM resources, referred to as vertical flavour $f \in N^v_{v-flv}$. Each flavour $f \in N^v_{flv}$ of the VNF $v \in N_{vnf}$, where $N_{vnf} = N_{upf} \cup N_{cpf} \cup N_{stf} \cup N_{app}$ has a certain amount of vCPU and vMEM $\omega^{vnf}_{c,m}(f)$ resources. These resources for applications VNFs N_{upf} and N_{app} are translated into a maximum amount of supportable traffic $\omega^{upf,app}_{t}(f)$, while for STFs N_{stf} and CPFs N_{cpf}, they are expressed in terms of a maximum number of supportable queries $\omega^{stf}_{q}(f)$ and signalling events $\omega^{cpf}_{e}(f)$ that they can handle, respectively. Table 6.1 summarizes the mobile network parameters.

UE Request Model. Both voice UEs and data UEs are considered. While the former is engaged in a voice call and, therefore, requires a call communication setup

TABLE 6.3: Binary decision variables.

Variable	Description
χ_g^u	Indicates if UE $u \in N_{ue}$ is associated with gNB $g \in N_{gnb}$.
$\chi_{f,n}^{\hat{v}}$	Indicates if VNF $\hat{v} \in N_{vnf}^{ue}(u)$ requested by UE u is mapped to the flavour $f \in N_{flv}^{v}$ of the same VNF on node $n \in N_{net}$.
$\chi_{f,n}^{v}$	Indicates if the flavour $f \in N_{flv}^{v}$ of the VNF $v \in N_{vnf}$ on node $n \in N_{net}$ has been used.

generating control plane events and queries towards, respectively, CPFs and STFs, the latter is using an application, which apart from communication with CPFs and STFs, requires also a data plane communication with the application service. All UE requests are modelled as *directed* graphs $G_{req} = (N_{req}, E_{req})$ where $N_{req} = N_{ue} \cup N_{vnf}^{ue}$ is the union of the set of UEs and the set of VNFs (i.e., UPFs, CPFs, STFs, APPs) that each UE has to have a connection with. E_{req} is the set of virtual links between UEs and their respective VNFs. Each data UE $u \in N_{ue}$ requires a certain amount of data rate $\omega_d^{ue}(u)$ for its application. Additionally, it is assumed that the UE communication setup process (e.g., during the initial UE to a gNB association, during the UE handover, etc.) for each UE generates a certain fixed amount of signaling events $\omega_e^{ue}(u)$ and queries $\omega_q^{ue}(u)$, which have to be handled by the CPFs and STFs, respectively.

6.4.1 ILP-based Method

Upon receiving UE requests, the MNO shall i) associate the UE with a gNB, ii) either place new VNFs on the computing nodes or use the already existing VNFs with/without scaling them, and iii) allocate enough computing resources to accommodate the request, if necessary. The goal is to satisfy the SFC requirements of the UE requests while at the same time making sure that the substrate network resources are used in an efficient manner. The SFC placement is modeled as a VNE problem, which is *NP–hard* and has been studied extensively in the literature [106], [112]. The embedding process consists of two parts: the node embedding and the link embedding. In the node embedding, each virtual node in the request (e.g., CPF, STF) is mapped to a substrate node (e.g., 5GE, 5GC). In the link embedding instead, each virtual link is mapped to a single substrate path. In both cases, the constraints of the nodes and links must be satisfied for an SFC mapping solution to be valid.

Before formulating the ILP model, for each UE, we first need to find the set of gNBs that provide coverage. Considering the location $loc(u)$ of the UE $u \in N_{ue}$ along with the location $loc(g)$ and the coverage radius $\delta(g)$ of gNB $g \in N_{gnb}$, the set of candidate gNBs $\Omega(u)$ for the UE u can be defined as follows:

$$\Omega(u) = \{g \in N_{gnb} | dis(loc(g), loc(u)) \leq \delta(g)\} \tag{6.1}$$

Formula (6.2) represents the objective functions considered in this ILP formulation. The first two arguments in (6.2) calculate the VNF deployment cost for, respectively, vertical and horizontal VNF scaling cases. The third argument takes into account the backhaul link usage cost, while the last one considers the PRB usage cost at the gNBs. ξ_c, ξ_m, ξ_e and ξ_p represent the cost of, respectively, a single vCPU core, 1Mb vMEM, 1Mbps backhaul bandwidth, and one PRB. Table 6.3 shows all binary variables used in this formulation.

Specifically, three objective functions are considered in this ILP formulation. While they pursue the same goal of minimizing the service provisioning cost for the MNO, they differ in terms of the used VNF scaling strategy, which is enforced by binary coefficients Λ_v and Λ_h. More specifically, if in formula (6.2), if $\Lambda_v = 1, \Lambda_h = 0$, then only vertical VNF scaling is considered in this objective, referred to as *VS*. If $\Lambda_v = 0, \Lambda_h = 1$ then only horizontal VNF scaling is considered in this objective, referred to as *HS*. Finally, if $\Lambda_v = 1, \Lambda_h = 1$ then both the vertical and horizontal VNF scaling strategies are considered in the objective function, referred to as *VHS* or a hybrid scaling, enabling the algorithm to pick the most appropriate VNF scaling strategy for a specific VNF based on a number of parameters such as VNF type and resource requirements.

$$\begin{aligned}
&\textit{Minimize}: \\
&\Lambda_v \sum_{v \in N_{vnf}} \sum_{f \in N^v_{v-flv}} \sum_{n \in N_{net}} (\xi_c \omega^{vnf}_c(f) + \xi_m \omega^{vnf}_m(f)) \chi^v_{f,n} + \\
&\Lambda_h \sum_{v \in N^{vnf}} \sum_{f \in N^v_{h-flv}} \sum_{n \in N_{net}} (\xi_c \omega^{vnf}_c(f) + \xi_m \omega^{vnf}_m(f)) \chi^v_{f,n} + \\
&\sum_{u \in N_{ue}} \sum_{\hat{e} \in E_{req(u)}} \sum_{e \in E_{net}} \xi_b \omega_d(u) \chi^{\hat{e}}_e + \sum_{u \in N_{ue}} \sum_{g \in N_{gnb}} \xi_p \omega_p(u) \chi^u_g
\end{aligned} \tag{6.2}$$

Regardless of the considered objective function (e.g., *VS*, *HS*, or *VHS*), all the following constraints have to be satisfied in order for an SFC placement solution to be valid. In the considered scenario, each UE $u \in N_{ue}$ has to be associated to only one gNB $g \in N_{gnb}$ (Constraint (6.3)), which belongs to the set of candidate gNBs $\Omega(u)$

of that UE (Constraint (6.4)) and has to have sufficient amount of PRBs in order to satisfy the data rate demand of all the UEs that are associated to that gNB (Constraint (6.5)).

$$\forall u \in N_{ue} : \sum_{g \in N_{gnb}} \chi_g^u = 1 \tag{6.3}$$

$$\forall u \in N_{ue} : \sum_{g \in N_{gnb} \setminus \Omega(u)} \chi_g^u = 0 \tag{6.4}$$

$$\forall g \in N_{gnb} : \sum_{u \in N_{ue}} \omega_{prb}^u \chi_g^u \leq \omega_{prb}^g \tag{6.5}$$

Since each VNF flavour $f \in N_{flv}^v$ has a certain amount of vCPU and vMEM resource requirement $\omega_{c,m}^{vnf}(f)$, each substrate network node $n \in N_{net}$, be it a 5GE or a 5GC, can host flavours of different VNF types (e.g., UPF, CPF, STF, application) as long as it has sufficient amount of vCPU and vMEM resources to host the VNF flavour (Constraint (6.6)).

$$\forall n \in N_{net} : \sum_{v \in N_{vnf}} \sum_{f \in N_{flv}^v} \omega_{c,m}^{vnf}(f) \chi_{f,n}^v \leq \omega_{c,m}(n) \tag{6.6}$$

The VNF flavours can serve the UEs as long as they have enough capacity (see Constraints (6.7), (6.8) and (6.8)). The VNF types are characterized by different sorts of resources. Specifically, while the UPF and application VNFs are characterized by a maximum amount of supportable traffic enforced by Constraint (6.7), the CPF VNF and the STF VNF are characterized by a maximum number of supported control plane events, respectively (see Constraint (6.8)) and queries (see Constraint (6.9)).

$$\begin{aligned} &\forall v \in N_{upf,app}, \hat{v} \in N_{vnf(v)}^{ue}, \forall f \in N_{flv}^v, \forall n \in N_{net} : \\ &\sum_{u \in N_{ue}} \omega_d^{ue}(u) \chi_{f,n}^{\hat{v}} \leq \omega_t^v(f) \end{aligned} \tag{6.7}$$

$$\begin{aligned} &\forall v \in N_{cpf}, \hat{v} \in N_{vnf(v)}^{ue}, \forall f \in N_{flv}^v, \forall n \in N_{net} : \\ &\sum_{u \in N_{ue}} \omega_e^{ue}(u) \chi_{f,n}^{\hat{v}} \leq \omega_e^v(f) \end{aligned} \tag{6.8}$$

$$\begin{aligned} &\forall v \in N_{stf}, \hat{v} \in N_{vnf(v)}^{ue}, \forall f \in N_{flv}^v, \forall n \in N_{net} : \\ &\sum_{u \in N_{ue}} \omega_q^{ue}(u) \chi_{f,n}^{\hat{v}} \leq \omega_q^v(f) \end{aligned} \tag{6.9}$$

Constraint (6.10) makes sure that each UE is connected to a single flavour of each VNF type that composes the UE's SFC request, while Constraint (6.11) guarantees that a VNF flavour is used if at least one UE uses it, where μ is a big number.

$$\forall u \in N_{ue}, \forall \hat{v} \in N^{ue}_{vnf}(u) : \sum_{n \in N_{net}} \sum_{f \in N^{\hat{v}}_{flv}} \chi^{\hat{v}}_{f,n} = 1 \tag{6.10}$$

$$\forall \hat{v} \in N^{ue}_{vnf}, \forall f \in N^{\hat{v}}_{flv}, \forall n \in N_{net} : \sum_{u \in N_{ue}} \chi^{\hat{v}}_{f,n} - \mu \chi^{v}_{f,n} \leq 0 \tag{6.11}$$

The backhaul link capacity constraint is handled by Constraint (6.12), which ensures that the virtual links can be mapped on a substrate backhaul link if the one has sufficient bandwidth to support the data rate demand of the virtual links. Lastly, Constraint (6.13) enforces for each virtual link to be a continuous path established between the gNB hosting the UE and the nodes hosting the VNFs of the SFC requested by the UE. $E^{\star i}_{net}$ is the set of the links that originate from any node and directly arrive at the node $i \in N_{net}$, while $E^{i\star}_{net}$ is the set of links that originates from the node i and arrive at any node directly connected to i.

$$\forall e \in E_{net} : \sum_{u \in N_{ue}} \sum_{\hat{e} \in E_{req(u)}} \omega^{ue}_{d}(\hat{e}) \chi^{\hat{e}}_{e} \leq \omega_b(e^{nm}) \tag{6.12}$$

$$\sum_{e \in E^{\star i}_{net}} \chi^{e^{n,m}}_{e} - \sum_{e \in E^{i\star}_{net}} \chi^{e^{n,m}}_{e} = \begin{cases} -1 & \text{if } i = n \\ 1 & \text{if } i = m \\ 0 & \text{otherwise} \end{cases} \tag{6.13}$$

$$\forall i \in N_{net}, \quad \forall e^{n,m} \in E_{req}$$

6.4.2 Heuristic

The ILP formulation becomes computationally intractable as the network's size increases (e.g., the number of gNBs, the number of VNFs, and the number of UEs). For example, it takes a day on an Intel Core i7 laptop (3.0 GHz CPU, 16 Gb RAM) using Gurobi solver [15] to associate and serve 50 UEs making service requests, each composed of 4 VNFs in a network composed of 4 gNBs and a core. To tackle the scalability issue of the ILP formulation, we propose a heuristic (the pseudo code is not shown due to space limitation) able to find near-optimal solutions for all the requests in a considerably shorter time.

Algorithm 3: *SmartScale*

Input: (G_{net}, G_{req})
Output: UEs association, SFC placement, VNF scaling, and resource allocation ;

Phase 1: Find candidate gNBs for each UE;
for $u \in N_{ue}$ **do**
  $cand\_gnb(u) \leftarrow \emptyset$;
  for $g \in N_{gnb}$ **do**
    $\omega^{ue}_{prb}(u,g) \leftarrow Calc\_prb(u,g)$;
    if $\omega^{ue}_{prb}(u,g) \leq \omega_{prb}(g)$ **then**
      $cand\_gnb(u) \leftarrow g$;

Phase 2: Check if already there is a VNF in the network to serve the UE;
for $u \in N_{ue}$ **do**
  for $v \in N^{ue}_{vnf}(u)$ **do**
    $map\_cost \leftarrow +\infty$;
    $tmp\_cost \leftarrow 0$;
    $tmp\_flv = tmp\_n = tmp\_gnb \leftarrow \emptyset$;
    $flag \leftarrow false$;
    for $n \in N$ **do**
      for $f \in N_{flv}$ **do**
        if $N^{v}_{f,n} == 1$ **then**
          for $g \in cand\_gnb(u)$ **do**
            if $\omega^{ue}_{d}(u) \leq \omega_b(e^{gn})$ **then**
              • compute link and prb cost for u;
              $tmp\_cost \leftarrow \xi^{e}_{g,n} + \xi^{prb}_{g}$
            if $tmp\_cost < map\_cost$ **then**
              $map\_cost \leftarrow tmp\_cost$;
              $tmp\_flv \leftarrow f$;
              $tmp\_n \leftarrow n$;
              $tmp\_gnb \leftarrow g$;
              $flag \leftarrow true$;
    if $flag == true$ **then**
      • associate u to tmp_gnb;
      • allocate tmp_flv on node tmp_n to the u;
      • construct the forwarding graph for each UE;
      • update prb resources and flavor capacity;
      • update the link resources;
  Phase 3: Associate the UE to a gNB, instantiate a new VNF on a node and allocate resources;
  if $flag == false$ **then**
    for $i \in weight \downarrow$ **do**
      $map\_cost \leftarrow +\infty$;
      for $g \in cand\_gnb(u)$ **do**
        if $\omega^{ue}_{d}(u) \leq \omega_b(e^{gn})$ ***and*** $i.(c,m) \leq \omega_{c,m}(n)$ **then**
          • compute cpu, mem, link, and prb cost for u;
          $tmp\_cost \leftarrow \xi^{e}_{i.(g,n)} + \xi^{prb}_{i.g} + \xi_{i.(c,m)}(i.n)$
        if $tmp\_cost < map\_cost$ **then**
          $map\_cost \leftarrow tmp\_cost$;
          $tmp\_flv \leftarrow i.flv$;
          $tmp\_n \leftarrow i.n$;
          $tmp\_gnb \leftarrow i.g$;
          $flag \leftarrow true$;
      if $flag == true$ **then**
        • associate u to tmp_gnb;
        • embed tmp_flv on node tmp_n;
        • allocate tmp_flv on node tmp_n to the u;
        • update cpu, mem, prb, and flavor capacity;
        • construct the forwarding graph for each UE;
        • update the link resources;

The proposed heuristic algorithm (3) pursues the hybrid VNF scaling objective of the ILP formulation and consists of three phases. The algorithm iterates over all the gNBs to find candidate gNBs for each UE in the first phase. A gNB is considered as a candidate if the UE is under the coverage radius of the gNB, and the gNB has a sufficient amount of PRBs to satisfy the UE's data rate demand.

In the second phase, the algorithm tries to serve the UEs from the existing VNFs in the network. The algorithm begins to serve UEs in sequence by looping over all the UEs, the candidate gNBs of the UE, and all the computing nodes to find if an instance of the UE's requested VNFs exists on the node. If yes, then the cost of serving that VNF instance will be computed. The cost encompasses the cost of PRB resources to associate the UE with the candidate gNB and the link resources that are needed to make a continuous path from the UE to the VNF instance. It is worth noting that this phase does not take into account the cost of using CPU and memory resources since the VNF instance already exists on the node, and there is no need to allocate computing resources to embed the VNF. After computing the cost for each possible solution (i.e., VNF, nodes, gNB) and finding a solution with a minimum cost, the VNF instance will be allocated to the UE, the UE will be associated with a gNB, and a path will be established from the gNB that the UE is associated with to the node hosting the VNF. This is followed by updating the network resources.

The third phase of the algorithm attempts to accommodate those UE requests for which there was no preexisting candidate VNF in the second phase. Thus in this phase, the algorithm tries to instantiate a new VNF instance of the requested service. Like the ILP, a weighting factor is considered for computing the cost of all the different flavours of each VNFs. In this regard, the algorithm sorts the solutions based on the cost in ascending order and loops over all the solutions until reaching a case that leads to minimum cost. Unlike the second phase, the solution cost in this phase encompasses the computing resources cost, link cost, and PRB cost. After finding a flavour of the VNF instance that complies with the node, link, and PRB resource demand, the VNF flavour will be embedded on the node, and the resources will be allocated then updated.

6.5 Performance Evaluation

The goal of this section is to compare the presented ILP-based and heuristic algorithms. We shall first describe the simulation setup used in our study. We will then

discuss the outcomes of the numerical simulations carried out in Python using Gurobi mathematical optimization solver [15].

6.5.1 Simulation Environment

A mobile network composed of 5 nodes is considered in this work, out of which one is the core, and the rest are gNBs. The gNBs are connected to the core node through 10 Gbps backhaul links. Both the core node and the gNBs have collocated MEC servers that possess, respectively, 12 and 4 CPU cores and 12 and 4-gigabyte memory, and are endowed with virtualization capability. Each CPU core is assumed to be equipped with 1.5 GHz clock rate. We assume four VNF categories, UPF, CPF, STF, and applications, with the last being in 5 types differentiated by their provided services and resource boundness (i.e., CPU-bound, memory-bound, or CPU-memory-bound). Each VNF instance can be shared among multiple UEs as long as it has sufficient capacity. Moreover, each VNF is available in multiple flavors, which defines the combination of CPU and memory resources allocated to that VNF. A VNF instance can have at least/most one/three CPU core(s) and one/three GB of memory. Thus, 9 VNF flavors exist for each VNF category.

Two different types of mobile UEs are considered in this work, as already introduced in Section 6.4. The first type of UEs are data UEs that make data requests, and they use the application VNFs existing in the system. The second type of UEs is voice UEs that do not ask for application but use voice services in the network and generate control messages. The purpose of considering voice UEs in the system is to show the impact of having different request types with diverse requirements and increasing the load on STF and CPF components to trigger scaling operation. The UE requests arrive sequentially in batches, each composed of 5 requests. It is assumed that with the arrival of a new data UE batch, there are four voice UE batches and that the UEs from the previous batches change their locations by moving in random direction with speed selected from the set $\{5, 25, 50\}$km/h, mimicking pedestrians, cyclist, and cars, while still keeping their data rate requirements. We consider up to 10 batches of data UE (50 data UEs in total) intending to trigger both scale-up and scale-out operations. After 50 data UEs, we gradually decrease the number of data UEs by 5 to trigger the scale-down and scale-in operations, if necessary, releasing the allocated resources.

As mentioned in Section 6.4, the capacity of CPF and STF VNFs are characterized, respectively, in terms of a maximum number of supportable events and

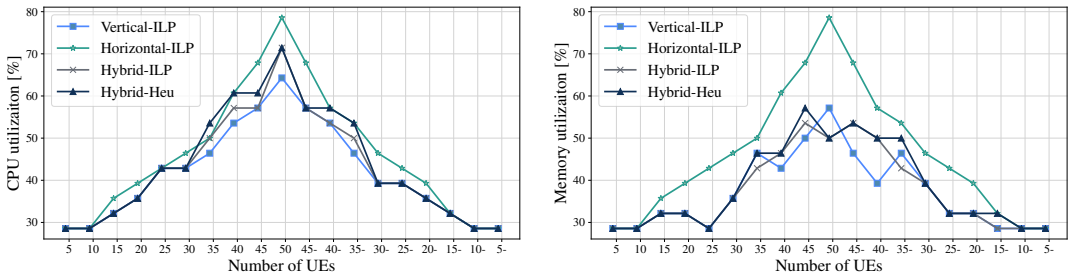
queries, while UPF and application VNFs are characterized in terms of throughput. All of these metrics are derived from the CPU and memory resource of the VNF. Specifically, it is assumed that the throughput, event capacity, and query capacity of, respectively, UPF, CPF, and STF are 80% dependent on the CPU and 20% on the memory. The capacity of CPF and STF VNFs has been computed following the approach in [113]. The CPU contribution in the overall capacity of a VNF is computed as the number of cores multiplied by each core's clock rate divided by the number of clocks required to process one bit of data (considered 10 in our scenario). Besides, if an application VNF is CPU-bound, the throughput depends on the CPU and vice versa. The same approach applies to the memory-bound application VNFs. In the case of CPU-memory-bound application VNF, instead, the throughput is equally dependent on the CPU and memory of that VNF.

6.5.2 Simulation Results

The reported results are the average of 5 simulations with 95% confidence intervals.

Resource utilization. Figure 6.5 illustrates the CPU and the memory utilization of the nodes (for a single simulation run) together with the under-utilization of the VNFs as a function of the number of UE requests for both ILP-based and heuristic VNF scaling algorithms. As expected, the vertical and the horizontal VNF scaling strategies achieve, respectively, the lowest and the highest CPU and memory utilization at the computing nodes (e.g., 5GEs, 5GC) as shown in Fig. 6.5a and Fig. 6.5b. This is because the horizontal VNF scaling strategy instantiates more VNFs of the same type allocating both CPU and memory resources from the host node even if the scaling is triggered due to the lack of only CPU or memory resource. Conversely, the vertical scaling strategy, thanks to its ability to resize the VNFs according to the need of having more/less CPU, or memory, or both, uses the node resources more efficiently while requiring the least amount of CPU and memory. For the hybrid VNF scaling strategy, we observe that the ILP-based and heuristic algorithms' performance resembles, and their CPU and memory utilization in most cases lies in between the ones achieved by the vertical and horizontal scaling approaches. This is justified by the fact that in this case, depending on the need, both vertical and horizontal VNF scalings are performed, as shown in Fig. 6.7.

The total under-utilization of the VNFs is computed based on the usage of throughput for UPF and application VNFs, queries for the STF VNFs, and control plane event for the CPF VNFs derived from the CPU and memory of the VNFs. As



(A) CPU utilization. (B) Memory utilization.

FIGURE 6.5: CPU and memory utilization of the nodes for the VNF scaling algorithms.

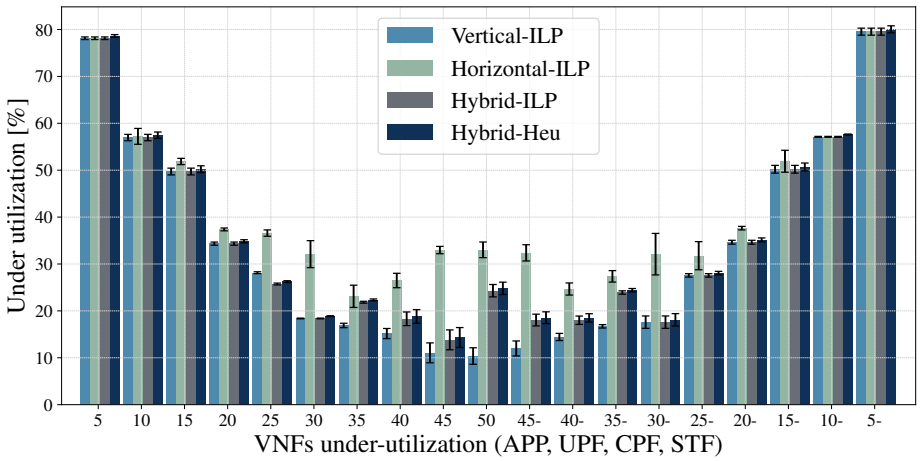


FIGURE 6.6: Under-utilization of VNFs for the VNF scaling algorithms.

expected, it reduces with the increase in the number of UE requests (see Fig. 6.6). While the horizontal VNF scaling strategy consumes the highest amount of CPU and memory of the nodes to instantiate VNFs, those VNFs and, therefore, those resources are not used efficiently, leading to the highest total VNF under-utilization. As opposed to the horizontal VNF scaling, the vertical VNF scaling strategy demonstrates the lowest total VNF under-utilization, leading to the most optimal utilization of the VNFs. As for the performance of the hybrid VNF scaling strategies, it is very similar to that of the vertical scaling with a slightly higher VNF under-utilization. It can also be observed that the difference between the VNF under-utilization achieved by the scalings strategies is more evident when the number of UE requests is high. This is because the more is the UEs, the more is the traffic demand, and, therefore, the more are the number of VNF scalings.

Number of VNF scalings. The number of different types of VNF scalings (e.g., scale-up, down, in, out) for the considered ILP-based and heuristic VNF scaling algorithms for varying numbers of UEs is shown in Fig. 6.7. As expected, more VNF scaling operations are induced when the number of UEs increases in the network. We can observe that mostly VNF scale-up and scale-out operations perform when

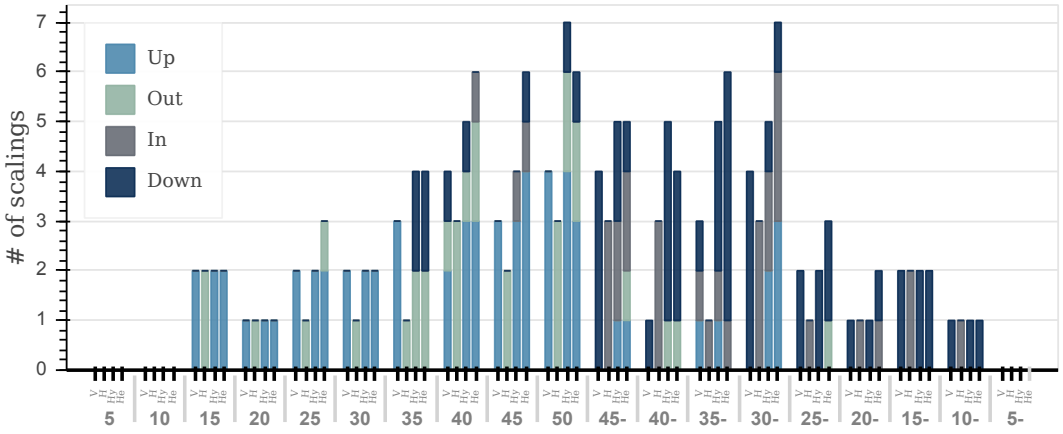
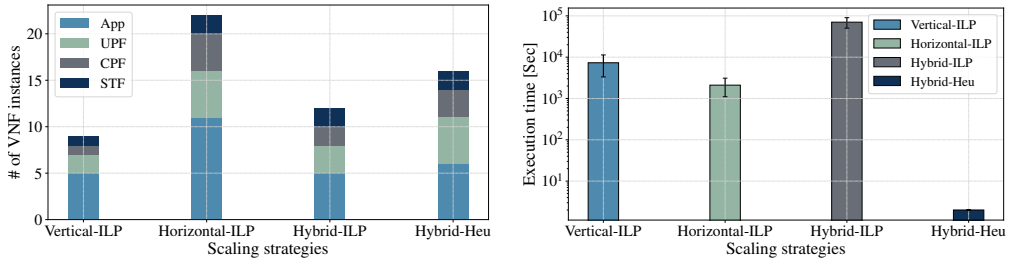


FIGURE 6.7: Number of VNF scalings.



(A) Number of VNF instances. (B) Execution time.

FIGURE 6.8: Number of different VNF instances, their scaling types, and the execution time for the VNF scaling algorithms.

the number of UEs increases, with most of the scale-ups/outs being triggered in the case of vertical/horizontal VNF scaling. On the other hand, when the UEs start leaving the network, VNF scale-up and scale-out operations are more dominant. It is worth mentioning that in some rare cases, VNF scale-down and scale-in operations are triggered even if the number of UEs increase in the network, while sometimes VNF scale-up and scale-out are performed when the UEs leave the network. This is due to the ability of the proposed algorithms to perform a customized VNF scaling. For instance, when the scaling-up of a VNF is needed, it might be more efficient to increase the CPU resource and decrease the allocated memory in order to meet the request demand and, at the same time, minimize the provisioning cost and vice versa.

Number of VNF instances. Figure 6.8a illustrates the average number of VNF instances for all VNF categories for the considered scaling strategies after embedding 50 UE requests. It can be observed that in both the ILP-based and heuristic VNF scaling strategies, there are a way more application VNFs than the other VNF categories, and among the application VNFs, the highest number of VNFs are instantiated by the horizontal VNF scaling, as expected. As for the UPF, CPF, and STF VNFs, there are fewer instances of them due to the fact that these VNFs have

much higher capacity, resulting in less frequent scalings. Naturally, the total number of VNF instances of the hybrid VNF scaling strategies lies in between the ones of the vertical and horizontal strategies due to being able to perform both vertical and horizontal VNF scaling.

Execution time. The main motivation for proposing the heuristic algorithm for the hybrid VNF scaling strategy is to address the scalability issue of the ILP-based algorithms. Figure 6.8 shows the average execution time for all the algorithms for associating 50 UE to the network, embedding the SFC requests, and performing VNF scaling. It can be observed that the execution time of the heuristic algorithm is at least three orders of magnitude less than that of the ILP-based algorithms, making it applicable in more practical scenarios and more suitable for various 5G use cases. Thus, the heuristic algorithm is much more scalable compared to ILP-based algorithms. Nonetheless, this comes at the expense of sub-optimal mapping solutions, leading to a slightly lower performance compared to its ILP-based counterpart.

6.6 Discussion

In this work, we studied a joint UE association, SFC placement, VNF scaling problem in the scenario of an end-to-end 5G network employing ILP and heuristic algorithms. Specifically, vertical, horizontal, and hybrid VNF scaling strategies have been compared, and their trade-offs are analyzed. We demonstrated that while the vertical VNF scaling is the most efficient strategy in utilizing the resources of the nodes and VNFs, it does not provide high availability to those VNFs due to their fewer instances compared to the horizontal VNF scaling strategy. However, the high availability of the horizontal VNF scaling strategy came at the expense of high CPU and memory usage of the nodes and high VNF under-utilization, making it inefficient from the resource utilization perspective. The hybrid VNF scaling strategy, on the other hand, exhibited a better compromise between the high availability of the VNFs and the resource utilization of the nodes and the VNFs.

Chapter 7

Conclusion and Future Work

This chapter summarizes the research presented in this dissertation, concluding observations and suggests some promising directions for future work.

7.1 Conclusion

5G is on the horizon with the promise to revolutionize the mobile communication landscape. Unlike the previous generations of mobile communication technologies that solely focused on improving the network capacity and increasing the data transmission rate, 5G makes a substantial transformation by focusing on three main pillars, each targeting demands for specific use cases. Firstly, similar to its predecessor 4G, 5G devotes enormous attention to increasing the capacity of the network, which enables many applications such as high-quality video streaming, VR/AR, online gaming, and many other bandwidth-hungry applications. Secondly, 5G intends to enable a massive number of IoT devices to join and communicate through the network. The ability to support a vast number of devices facilitates many applications such as smart cities and smart homes. Finally, 5G devotes a considerable attention to the latency reduction in order to deliver sub-millisecond latency and lift the barriers towards implementing applications such as autonomous driving, e-health, and industrial automation.

MEC along with NFV have the potential to meet the requirement of the 5G networks in terms of latency mitigation, flexibility enhancement, and CapEx and OpEx reduction. MEC technology aims to shift the network intelligence, processing, storage, and virtualization capabilities to the edge of the network in the proximity of the end-users. MEC significantly reduces the time needed to access the content;

therefore, it considerably contributes to the overall E2E latency reduction. In the context of mobile networks, MEC can be employed to host NFs and applications deployed as VNFs. Scarcity of resources, resource provisioning cost, and heterogeneity of resources are the main challenges towards realizing MEC technology in mobile networks.

In this doctoral dissertation, we investigated different strategies for content caching/prefetching, SFC placement, and VNF lifecycle management in the scenario of MEC-enabled 5G networks. First, we investigated the problem of video content prefetching/caching in MEC-enabled mobile networks. We specifically studied the trade-offs between different prefetching strategies, their benefits, and challenges with the ultimate goal of providing a set of methods for MNOs for video content prefetching. Second, we investigated the joint problem of user association and VNF placement in MEC-enabled 5G networks. We specifically proposed several algorithms for the efficient placement of SFCs on the substrate network while respecting the data rate and latency demands of the UEs. Finally, we explored the trade-off between different scaling strategies and proposed a scaling approach that attempts to minimize the cost for MNOs while respecting the Service Level Agreement (SLA) for the UEs.

In Chapter 4, we addressed the problem of limited cache storage in MEC servers in the context of mobile networks for the online DASH video streaming scenario. Specifically, we proposed a novel ML-driven predictive prefetching method for the problem of DASH video streaming in MEC-enabled mobile networks. Our first task was to design prediction algorithms that can predict the number of segment requests, bitrate of the segments, and gNB association of the UE in a prediction time window. In this regard, three prediction algorithms were proposed that showed a high accuracy of (83-88-99%) for the three predictive tasks. The next step was to devise a prefetching algorithm that is able to make a trade-off between the number of UEs served from the edge and the resource usage at each of the MEC nodes. An ILP model with two objectives was proposed to reach an optimal solution for the video content prefetching and transcoding at the edge, followed by a heuristic algorithm that achieves a near-optimal solution in an exceedingly shorter time scale. We were able to attain a MEC cache-hit ratio of 60%, which demonstrates that we achieved a reduction of the access delay for 60% of the requests. We demonstrated that our proposed algorithms could reduce the BH link utilization to a very large extend through caching at the edge by using the max byte-hit objective in a live streaming scenario with segment request overlaps.

In Chapter 5, we studied the joint problems of user association, SFC placement, and resource allocation in a hierarchical MEC-enabled 5G network setup. We first provided a comprehensive E2E delay estimation model for users, taking into account the transmission and the propagation time over the air and the transport links along with the VNF processing time. Next, aiming at associating UEs to the gNBs and embedding the services on the substrate network, we employed MILP techniques to provide a novel formulation of the problem, with the objective to minimize the service provisioning cost, the impact of VNF migration on the QoE of the UEs, and the transport network utilization. Having the goal to tackle the scalability issue of the proposed MILP method, a heuristic algorithm was proposed to reach a near-optimal solution in order to minimize the impact of VNF migration on the perceived QoE by the UEs in a much shorter time. Comprehensive simulations demonstrated a comparison between the proposed algorithms by considering different types of service requests with diverse data rates and E2E latency demands.

The dynamic nature of mobile networks demands algorithms that can actively adjust to the changes of mobile networks. In order for MNOs to be competitive in the market, ensuring optimal resource utilization, and lowering the service provisioning cost, scaling approaches are needed that can dynamically adjust resource allocation in order to have an optimal resource utilization while meeting user demands. In this regard, in Chapter 6 we studied the joint UE association, SFC placement, and VNF scaling problem in the scenario of an E2E 5G network. Specifically, we conducted an extensive comparison of different scaling strategies, namely vertical, horizontal, and hybrid scaling, and provided analysis of trade-offs. We used ILP models with three objectives, each trying to minimize the cost for the given scaling strategy (vertical, horizontal, hybrid). We proposed a heuristic algorithm following the same objective of the hybrid scaling strategy. We demonstrated that while the vertical VNF scaling is the most efficient strategy in utilizing the resources of the nodes and VNFs, it does not provide high availability to those VNFs due to their fewer instances compared to the horizontal VNF scaling strategy. However, the high availability of the horizontal VNF scaling strategy is achieved at the expense of high CPU and memory usage of the nodes and high VNF under-utilization, making it inefficient from the resource utilization perspective. The hybrid VNF scaling strategy, on the other hand, exhibited a better compromise between the high availability of the VNFs and the resource utilization of the nodes and VNFs.

7.2 Future Work

Inspired by the complexity of DASH video content prefetching, the importance of devising new approaches for predicting UE's requests, the need for proactive scaling of the network, and in order to study the problems in a realistic scenario, we have outlined the following future research directions to address the open issues arising in this dissertation adequately.

- As shown in Chapter 4, numerous factors, including RAN metrics, client metrics, and history of requests, are involved in the process of predicting user requests in the scenario of DASH video prefetching. In Chapter 4, we explored two major ML algorithms, namely Random Forest and Gradient Boosting Trees, which are shown to be effective in state-of-the-art works. One of the interesting research directions is to further investigate the problem of DASH video segment prediction, provide analysis of the auto-correlation of the dataset features, and study if the samples are independently distributed. Considering the high impact of the prediction phase on the overall prefetching performance and the negative effect of wrong predictions on both UE's perceived QoE and resource usage at the edge, we intend to achieve higher accuracy of the predictions. In this regard, we aim to further examine the performance of other prediction algorithms, such as Neural Networks (NN), and motivate our choice of the prediction algorithm based on data analysis.
- It has been previously emphasized that one of the main problems with the ILP prefetching algorithm is its execution time to reach the optimal solution. This drawback motivated us to devise a heuristic algorithm that reaches a near-optimal solution in a much shorter time scale. Thus, another intriguing research direction is to develop an E2E ML-based algorithm that, apart from predicting user requests, performs the prefetching process using a reinforcement learning agent. We aim to explore the applicability of replacement of the ILP and heuristic algorithms by an ML-based agent and provide a fully E2E automated system for the problem of DASH video content prefetching at the network edge.
- While we have already studied the trade-off between different scaling strategies (vertical, horizontal, and hybrid) in Chapter 6, an interesting research direction would be to extend our approach further by investigating the state exchange problem when scaling operation happens—this problem arises in the case of horizontal scaling for stateful applications and STF 5G core VNFs.

In this regard, we aim to extend the ILP formulation and the heuristic algorithm by including the cost of exchanging the states related to the UEs. Moreover, we are interested in showing the applicability of our proposed approaches by confirming their performance through a Proof of Concept (PoC) implementation.

- Considering the fact that our approach in Chapter 6 is reactive and tries to adapt the network based on the issued requests, users might be affected for a duration of time until the system adapts itself. Therefore, one interesting research direction is to devise a method to adjust the system before the users issue their request. In this regard, as future work, we aim to employ ML techniques to predict the number of users who join the network and their service requirements. Knowing future requests allows us to scale the applications and 5G core components before they being requested by the UE, consequently avoiding QoE degradation caused by reconfiguration.
- Finally, regarding the combinatorial optimization techniques used for problem formulation in this theses, it would be of great interest for us to investigate Optimization Modulo Theories (OMT) techniques to formalize and model the studied problems, drawing a comparison with the ILP/MILP techniques in terms of their execution time and the optimality of mapping solutions.

Bibliography

[1] GSA, "The Road to 5G: Drivers, Applications, Requirements and Technical Development," *Global Mobile Suppliers Association*, 2015.

[2] Cisco, "Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2017–2022," 2019.

[3] M Series, "IMT Vision–Framework and Overall Objectives of the Future Development of IMT for 2020 and beyond," *Recommendation ITU*, 2015.

[4] J. G. Andrews, S. Buzzi, W. Choi, S. V. Hanly, A. Lozano, A. C. Soong, and J. C. Zhang, "What Will 5G be?" *IEEE Journal on selected areas in communications*, vol. 32, no. 6, pp. 1065–1082, 2014.

[5] ETSI, "GR MEC 001 V2.1.1," *Multi-access Edge Computing (MEC); Terminology*, 2019.

[6] ——, "GS NFV 002 V1.2.1," *Network Functions Virtualisation (NFV); Architectural Framework*, 2014.

[7] ——, "GR MEC 017 V1.1.1," *Mobile Edge Computing (MEC); Deployment of Mobile Edge Computing in an NFV environment*, 2018.

[8] ——, "GS NFV-MAN 001," *Network Functions Virtualization (NFV); Management and Orchestration*, 2014.

[9] ——, "GS MEC 002 V1.1.1," *Mobile Edge Computing (MEC); Technical Requirements*, 2016.

[10] S. Abdelwahab, B. Hamdaoui, M. Guizani, and T. Znati, "Network Function Virtualization in 5G," *IEEE Communications Magazine*, vol. 54, no. 4, pp. 84–91, 2016.

[11] H. Wu, F. Zhou, Y. Chen, and R. Zhang, "On Virtual Network Embedding: Paths and Cycles," *IEEE Transactions on Network and Service Management*, vol. 17, no. 3, pp. 1487–1500, 2020.

[12] Z. Despotovic, A. Hecker, A. N. Malik, R. Guerzoni, I. Vaishnavi, R. Trivisonno, and S. A. Beker, "VNetMapper: A Fast and Scalable Approach to Virtual Networks Embedding," in *Proc of IEEE ICCCN*, Shanghai, China, 2014, pp. 1–6.

[13] M. Chowdhury, M. R. Rahman, and R. Boutaba, "Vineyard: Virtual Network Embedding Algorithms with Coordinated Node and Link Mapping," *IEEE/ACM Transactions on Networking*, vol. 20, no. 1, pp. 206–219, 2011.

[14] S. Zhang, Z. Qian, J. Wu, S. Lu, and L. Epstein, "Virtual Network Embedding with Opportunistic Resource Sharing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 25, no. 3, pp. 816–827, 2013.

[15] *Gurobi mathematical optimization solver*, Accessed on 20.12.2020. [Online]. Available: https://www.gurobi.com/.

[16] K. Liang, J. Hao, R. Zimmermann, and D. K. Yau, "Integrated Prefetching and Caching for Adaptive Video Streaming over HTTP: an Online Approach," in *Proc. of ACM MM*, Portland, Oregon, USA, 2015.

[17] R. Behravesh, D. F. Perez-Ramirez, A. Rao, D. Harutyunyan, R. Riggio, and R. Steinert, "ML-Driven DASH Content Pre-Fetching in MEC-Enabled Mobile Networks," in *Proc. of IEEE CNSM*, Izmir, Turkey, 2020, pp. 1–7.

[18] R. Behravesh, E. Coronado, D. Harutyunyan, and R. Riggio, "Joint User Association and VNF Placement for Latency Sensitive Applications in 5G Networks," in *Proc. of IEEE CloudNet*, Coimbra, Portugal, 2019.

[19] D. Harutyunyan, R. Behravesh, and N. Slamnik-Krijestorac, "Cost-Efficient Placement and Scaling of 5G Core Network and MEC-Enabled Application VNFs," in *Proc. of IEEE IM*, Bordeaux, France, 2021.

[20] F. Z. Yousaf, M. Bredel, S. Schaller, and F. Schneider, "NFV and SDN — Key Technology Enablers for 5G Networks," *IEEE Journal on Selected Areas in Communications*, vol. 35, no. 11, pp. 2468–2478, 2017.

[21] V.-G. Nguyen, A. Brunstrom, K.-J. Grinnemo, and J. Taheri, "SDN/NFV-based Mobile Packet Core Network Architectures: A Survey," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 3, pp. 1567–1602, 2017.

[22] B. A. A. Nunes, M. Mendonca, X.-N. Nguyen, K. Obraczka, and T. Turletti, "A Survey of Software-Defined Networking: Past, Present, and Future of Programmable Networks," *IEEE Communications surveys & tutorials*, vol. 16, no. 3, pp. 1617–1634, 2014.

[23] D. Kreutz, F. M. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig, "Software-Defined Networking: A Comprehensive Survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14–76, 2015.

[24] S. H. Yeganeh, A. Tootoonchian, and Y. Ganjali, "On Scalability of Software-Defined Networking," *IEEE Communications Magazine*, vol. 51, no. 2, pp. 136–141, 2013.

[25] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "OpenFlow: Enabling Innovation in Campus Networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, 2008.

[26] *Open networking foundation*, `https://www.opennetworking.org/`, Accessed on 20.12.2018.

[27] B. Pfaff and B. Davie, "The Open vSwitch Database Management Protocol," Tech. Rep., 2013.

[28] A. Doria, J. H. Salim, R. Haas, H. Khosravi, W. Wang, L. Dong, R. Gopal, and J. Halpern, "Forwarding and Control Element Separation (ForCES) Protocol Specification," Tech. Rep., 2010.

[29] A. Lara, A. Kolasani, and B. Ramamurthy, "Network Innovation Using OpenFlow: A Survey," *IEEE communications surveys & tutorials*, vol. 16, no. 1, pp. 493–512, 2014.

[30] Y. Jarraya, T. Madi, and M. Debbabi, "A Survey and a Layered Taxonomy of Software-Defined Networking," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 4, pp. 1955–1980, 2014.

[31] G. ETSI, "Nfv 001 v1.2.1," *Network Functions Virtualisation (NFV), Use Cases*, 2017.

[32] R. Behravesh, E. Coronado, and R. Riggio, "Performance evaluation on virtualization technologies for nfv deployment in 5g networks," in *Proc. of IEEE NetSoft*, 2019, pp. 24–29.

[33] ETSI, "ETSI 123 501 V15.9.0," *5G, System Architecture for the 5G System (5GS)*, 2020.

[34] S. Clayman, E. Maini, A. Galis, A. Manzalini, and N. Mazzocca, "The Dynamic Placement of Virtual Network Functions," in *Proc. of IEEE NOMS*, Krakow, Poland, 2014, pp. 1–9.

[35] J. Xia, Z. Cai, and M. Xu, "Optimized Virtual Network Functions Migration for NFV," in *Proc. of IEEE ICPADS*, Wuhan, China, 2016.

[36] P. Mell, T. Grance, *et al.*, "The NIST Definition of Cloud Computing," Tech. Rep., 2011.

[37] R. Mijumbi, J. Serrat, J.-L. Gorricho, N. Bouten, F. De Turck, and R. Boutaba, "Network Function Virtualization: State-of-the-art and Research Challenges," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 236–262, 2016.

[38] A. Bentaleb, B. Taani, A. C. Begen, C. Timmerer, and R. Zimmermann, "A Survey on Bitrate Adaptation Schemes for Streaming Media over HTTP," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 1, pp. 562–585, 2018.

[39] IEC_MPEG, "Information Technology Dynamic Adaptive Streaming over HTTP (DASH)-part 1: Media Presentation Description and Segment Formats," Tech. Rep., 2012.

[40] 3GPP, "Transparent end-to-end packet switched streaming service (PSS) (release 9)," Tech. Rep., 2009.

[41] ——, "Progressive Download and Dynamic Adaptive Streaming over HTTP (release 10)," Tech. Rep., 2010.

[42] M. Seufert, S. Egger, M. Slanina, T. Zinner, T. Hoßfeld, and P. Tran-Gia, "A Survey on Quality of Experience of HTTP Adaptive Streaming," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 1, pp. 469–492, 2014.

[43] P. Juluri and D. Medhi, "Cache'n DASH: Efficient Caching for DASH," *ACM SIGCOMM Computer Communication Review*, vol. 45, no. 4, pp. 599–600, 2015.

[44] A. Araldo, F. Martignon, and D. Rossi, "Representation Selection Problem: Optimizing Video Delivery Through Caching," in *Proc. of IEEE IFIP*, Vienna, Austria, 2016.

[45] Y. Li, P. A. Frangoudis, Y. Hadjadj-Aoul, and P. Bertin, "A Mobile Edge Computing-based Architecture for Improved Adaptive HTTP Video Delivery," in *Proc. of IEEE CSCN*, Berlin, Germany, 2016.

[46] Y. Li, P. A. Frangoudis, Y. Hadjadj-Aoul, and P. Bertin, "A Mobile Edge Computing-assisted Video Delivery Architecture for Wireless Heterogeneous Networks," in *Proc. of IEEE ISCC*, Heraklion, Greece, 2017.

[47] Y. Tan, C. Han, M. Luo, X. Zhou, and X. Zhang, "Radio Network-aware Edge Caching for Video Delivery in MEC-Enabled Cellular Networks," in *Proc. of IEEE WCNCW*, Barcelona, Spain, 2018.

[48] S. Kumar, D. S. Vineeth, *et al.*, "Edge Assisted DASH Video Caching Mechanism for Multi-access Edge Computing," in *Proc. of IEEE ANTS*, Indore, India, 2018.

[49] S. K. Mehr, P. Juluri, M. Maddumala, and D. Medhi, "An Adaptation-aware Hybrid Client-cache Approach for Video Delivery with Dynamic Adaptive Streaming over HTTP," in *Proc. of IEEE/IFIP NOMS*, Taipei, Taiwan, 2018.

[50] W. Shi, Q. Li, C. Wang, G. Shen, W. Li, Y. Wu, and Y. Jiang, "LEAP: Learning-based Smart Edge with Caching and Prefetching for Adaptive Video Streaming," in *Proc. of ACM IWQOS*, Phoenix Arizona, USA, 2019.

[51] A. Samba, Y. Busnel, A. Blanc, P. Dooze, and G. Simon, "Instantaneous Throughput Prediction in Cellular Networks: Which Information is Needed?" In *Proc. of IFIP/IEEE IM*, Lisbon, Portugal, 2017.

[52] M. Karimzadeh, Z. Zhao, L. Hendriks, R. D. O. Schmidt, S. La Fleur, H. Van Den Berg, A. Pras, T. Braun, and M. J. Corici, "Mobility and Bandwidth Prediction as a Service in Virtualized LTE Systems," in *Proc. of IEEE CloudNet*, Niagara Falls, ON, Canada, 2015.

[53] D. Raca, A. H. Zahran, C. J. Sreenan, R. K. Sinha, E. Halepovic, R. Jana, and V. Gopalakrishnan, "Back to the Future: Throughput Prediction for Cellular Networks Using Radio KPIs," in *Proc. of ACM MOBICOM*, New York, NY, USA, 2017.

[54] C. Yue, R. Jin, K. Suh, Y. Qin, B. Wang, and W. Wei, "LinkForecast: Cellular Link Bandwidth Prediction in LTE Networks," *IEEE Transactions on Mobile Computing*, vol. 17, no. 7, pp. 1582–1594, 2018.

[55] D. Raca, A. H. Zahran, C. J. Sreenan, R. K. Sinha, E. Halepovic, R. Jana, V. Gopalakrishnan, B. Bathula, and M. Varvello, "Incorporating Prediction into Adaptive Streaming Algorithms: A QoE Perspective," in *Proc. of ACM SIGCOM NOSSDAV*, New York, NY, USA, 2018.

[56] D. Raca, A. H. Zahran, C. J. Sreenan, R. K. Sinha, E. Halepovic, R. Jana, V. Gopalakrishnan, B. Bathula, and M. Varvello, "Empowering Video Players in Cellular: Throughput Prediction from Radio Network Measurements," in *Proc. of ACM MM*, New York, NY, USA, 2019.

[57] H. Mao, R. Netravali, and M. Alizadeh, "Neural Adaptive Video Streaming with Pensieve," in *Proc. of ACM SIGCOM COMM*, New York, NY, USA, 2017.

[58] D. Liu, L. Wang, Y. Chen, M. Elkashlan, K.-K. Wong, R. Schober, and L. Hanzo, "User Association in 5G Networks: A Survey and an Outlook," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 2, pp. 1018–1044, 2016.

[59] D. Liu, Y. Chen, K. K. Chai, and T. Zhang, "Nash Bargaining Solution based User Association Optimization in HetNets," in *Proc. of IEEE CCNC*, Las Vegas, NV, USA, 2014.

[60] Y. Lei, G. Zhu, C. Shen, Y. Xu, and X. Zhang, "Delay-aware User Association and Power Control for 5G Heterogeneous Network," *Mobile Networks and Applications*, vol. 24, pp. 1–13, 2018.

[61] M. Amine, A. Walid, A. Kobbane, and J. Ben-Othman, "New User Association Scheme Based on Multi-Objective Optimization for 5G Ultra-Dense Multi-RAT HetNets," in *Proc. of IEEE ICC*, Kansas City, MO, USA, 2018.

[62] A. S. Cacciapuoti, "Mobility-aware User Association for 5G mmWave Networks," *IEEE Access*, vol. 5, pp. 21 497–21 507, 2017.

[63] M. Amine, A. Kobbane, and J. Ben-Othman, "New Network Slicing Scheme for UE Association Solution in 5G Ultra Dense HetNets," in *Proc. of IEEE ICC*, Dublin, Ireland, 2020.

[64] S. Goyal, M. Mezzavilla, S. Rangan, S. Panwar, and M. Zorzi, "User association in 5g mmwave networks," in *Proc. of IEEE WCNC*, San Fran, CA, USA, 2017.

[65] D. Harutyunyan, A. Bradai, and R. Riggio, "Trade-offs in Cache-enabled Mobile Networks," in *Proc. of IEEE CNSM*, Rome, Italy, 2018.

[66] X. Ge, X. Li, H. Jin, J. Cheng, and V. C. Leung, "Joint User Association and User Sscheduling for Load Balancing in Heterogeneous Networks," *IEEE Transactions on Wireless Communications*, vol. 17, no. 5, pp. 3211–3225, 2018.

[67] N. Liakopoulos, G. Paschos, and T. Spyropoulos, "Robust User Association for Ultra Dense Networks," in *Proc. of IEEE INFOCOM*, Honolulu, HI, USA, 2018.

[68] A. Alleg, T. Ahmed, M. Mosbah, R. Riggio, and R. Boutaba, "Delay-aware VNF Placement and Chaining based on a Flexible Resource Allocation Approach," in *Proc. of IEEE CNSM*, Tokyo, Japan, 2017.

[69] Q. Zhang, F. Liu, and C. Zeng, "Adaptive Interference-aware VNF Placement for Service-customized 5G Network Slices," in *Proc. of IEEE INFOCOM*, Paris, France, 2019.

[70] S. Yang, F. Li, R. Yahyapour, and X. Fu, "Delay-sensitive and Availability-aware Virtual Network Function Scheduling for NFV," *IEEE Transactions on Services Computing*, 2019.

[71] M. M. Tajiki, S. Salsano, L. Chiaraviglio, M. Shojafar, and B. Akbari, "Joint Energy Efficient and QoS-Aware Path Allocation and VNF Placement for Service Function Chaining," *IEEE Transactions on Network and Service Management*, vol. 16, no. 1, pp. 374–388, 2019.

[72] L. Wang, Z. Lu, X. Wen, R. Knopp, and R. Gupta, "Joint Optimization of Service Function Chaining and Resource Allocation in Network Function Virtualization," *IEEE Access*, vol. 4, pp. 8084–8094, 2016.

[73] S. Agarwal, F. Malandrino, C.-F. Chiasserini, and S. De, "Joint VNF Placement and CPU Allocation in 5G," in *Proc. of IEEE INFOCOM*, Honolulu, HI, USA, 2018.

[74] Q. Zhang, Y. Xiao, F. Liu, J. C. Lui, J. Guo, and T. Wang, "Joint Optimization of Chain Placement and Request Scheduling for Network Function Virtualization," in *Proc. of IEEE ICDCS*, Atlanta, GA, USA, 2017.

[75] Y. Bi, C. Colman-Meixner, R. Wang, F. Meng, R. Nejabati, and D. Simeonidou, "Resource Allocation for Ultra-low Latency Virtual Network Services in Hierarchical 5G Network," in *Proc. of IEEE ICC*, Shanghai, China, 2019.

[76] D. Zhang, X. Lin, and X. Chen, "Multiple Instances Mapping of Service Function Chain with Parallel Virtual Network Functions," *Journal of Algorithms & Computational Technology*, vol. 13, 2019.

[77] H. Moens and F. De Turck, "VNF-P: A Model for Efficient Placement of Virtualized Network Functions," in *Proc. of IEEE CNS*, San Francisco, CA, USA, 2014.

[78] M. F. Bari, S. R. Chowdhury, R. Ahmed, and R. Boutaba, "On orchestrating virtual network functions," in *Proc. of IEEE CNSM*, Barcelona, Spain, 2015.

[79] M. Huang, W. Liang, Y. Ma, and S. Guo, "Throughput maximization of delay-sensitive request admissions via virtualized network function placements and migrations," in *Proc. of IEEE ICC*, Kansas City, MO, USA, 2018.

[80] N. Kiran, X. Liu, S. Wang, and C. Yin, "VNF Placement and Resource Allocation in SDN/NFV-Enabled MEC Networks," in *Proc. of IEEE WCNCW*, Seoul, Korea (South), 2020.

[81] A. Laghrissi and T. Taleb, "A survey on the placement of virtual resources and virtual network functions," *IEEE Communications Surveys Tutorials*, vol. 21, no. 2, pp. 1409–1434, 2019.

[82] J. Gil Herrera and J. F. Botero, "Resource Allocation in NFV: A Comprehensive Survey," *IEEE Transactions on Network and Service Management*, vol. 13, no. 3, pp. 518–532, 2016.

[83] Xin Li and Chen Qian, "A Survey of Network Function Placement," in *Proc. of IEEE CCNC*, Las Vegas, NV, USA, 2016.

[84] R. Cziva, C. Anagnostopoulos, and D. P. Pezaros, "Dynamic, Latency-Optimal VNF Placement at the Network Edge," in *Proc. of IEEE INFOCOM*, Honolulu, HI, USA, 2018.

[85] D. Harutyunyan, N. Shahriar, R. Boutaba, and R. Riggio, "Latency-Aware Service Function Chain Placement in 5G Mobile Networks," in *Proc. of IEEE NetSoft*, Paris, France, 2019.

[86] D. Cho, J. Taheri, A. Y. Zomaya, and P. Bouvry, "Real-time Virtual Network Function (VNF) Migration Toward Low Network Latency in Cloud Environments," in *Proc. of IEEE CLOUD*, Honolulu, HI, USA, 2017.

[87] F. Carpio, A. Jukan, and R. Pries, "Balancing the Migration of Virtual Network Functions with Replications in Data Centers," in *Proc. of IEEE/IFIP NOMS*, Taipei, Taiwan, 2018.

[88] H. Hawilo, M. Jammal, and A. Shami, "Orchestrating Network Function Virtualization Platform: Migration or Re-instantiation?" In *Proc. of IEEE CloudNet*, Prague, Czech Republic, 2017.

[89] I. Sarrigiannis, E. Kartsakli, K. Ramantas, A. Antonopoulos, and C. Verikoukis, "Application and Network VNF migration in a MEC-enabled 5G Architecture," in *Proc. IEEE CAMAD*, Barcelona, Spain, 2018.

[90] M. Sedaghat, F. Hernandez-Rodriguez, and E. Elmroth, "A Virtual Machine Re-packing Approach to the Horizontal vs. Vertical Elasticity Trade-off for Cloud Autoscaling," in *Proc. of ACM CAC*, Miami Florida, USA, 2013, pp. 1–10.

[91] K. Hwang, Y. Shi, and X. Bai, "Scale-out vs. Scale-up Techniques for Cloud Performance and Productivity," in *Proc. of IEEE CloudCom*, Singapore, 2014, pp. 763–768.

[92] W. Wang, L. Xu, and I. Gupta, "Scale Up vs. Scale Out In Cloud Storage and Graph Processing Systems," in *Proc. of IEEE IC2E*, Tempe, AZ, USA, 2015, pp. 428–433.

[93] T. V. K. Buyakar, A. K. Rangisetti, A. A. Franklin, and B. R. Tamma, "Auto Scaling of Data Plane VNFs in 5G Networks," in *Proc. of IEEE CNSM*, Tokyo, Japan, 2017, pp. 1–4.

[94] I. Alawe, Y. Hadjadj-Aoul, A. Ksentini, P. Bertin, and D. Darche, "On the Scalability of 5G Core network: the AMF Case," in *Proc. of IEEE CCNC*, Las Vegas, HI, USA, 2018, pp. 1–6.

[95] M. Moghaddassian, H. Bannazadeh, and A. Leon-Garcia, "Adaptive Auto-Scaling for Virtual Resources in Software-Defined Infrastructure," in *Proc. of IEEE IM*, Lisbon, Portugal, 2017, pp. 548–551.

[96] R. Gouareb, V. Friderikos, and A.-H. Aghvami, "Virtual Network Functions Routing and Placement for Edge Cloud Latency Minimization," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 10, pp. 2346–2357, 2018.

[97] H. Tang, D. Zhou, and D. Chen, "Dynamic Network Function Instance Scaling based on Traffic Forecasting and VNF Placement in Operator Data Centers," *IEEE Transactions on Parallel and Distributed Systems*, vol. 30, no. 3, pp. 530–543, 2018.

[98] T. Subramanya and R. Riggio, "Machine Learning-driven Scaling and Placement of Virtual Network Functions at the Network Edges," in *Proc. of IEEE NetSoft*, Paris, France, 2019, pp. 414–422.

[99] *Network Simulator ns-3*. [Online]. Available: `https://www.nsnam.org/`.

[100] *ns-3 generated dataset for DASH over dynamic radio access networks*. [Online]. Available: `{https://github.com/akhila-s-rao/ns3_dash_over_ran/}`.

[101] *An MPEG/DASH client-server ns3 module*, Accessed on 23.07.2020. [Online]. Available: `https://github.com/djvergad/dash`.

[102] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," in *Proc. of ACM SIGKDD*, New York, NY, USA, 2016.

[103] R. Caruana and A. Niculescu-Mizil, "An Empirical Comparison of Supervised Learning Algorithms," in *Proc. of ACM ICMLC*, New York, NY, USA, 2006.

[104] S. Kekki, W. Featherstone, Y. Fang, P. Kuure, A. Li, A. Ranjan, D. Purkayastha, F. Jiangping, D. Frydman, G. Verin, *et al.*, "MEC in 5G networks," *ETSI, White Paper*, 2018.

[105] M. Chowdhury, M. R. Rahman, and R. Boutaba, "Vineyard: Virtual Network Embedding Algorithms with Coordinated Node and Link Mapping," *IEEE/ACM Transactions on Networking*, vol. 20, no. 1, pp. 206–219, 2012.

[106] A. Fischer, J. F. Botero, M. T. Beck, H. De Meer, and X. Hesselbach, "Virtual Network Embedding: A Survey," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 4, pp. 1888–1906, 2013.

[107] A. Chiumento, M. Bennis, C. Desset, L. Van der Perre, and S. Pollin, "Adaptive CSI and Feedback Estimation in LTE and Beyond: a Gaussian Process Regression Approach," *EURASIP Journal on Wireless Communications and Networking*, vol. 2015, no. 1, p. 168, 2015.

[108] A. N. Toosi, J. Son, Q. Chi, and R. Buyya, "ElasticSFC: Auto-scaling Techniques for Elastic Service Function Chaining in Network Functions Virtualization-based Clouds," *Journal of Systems and Software*, vol. 152, pp. 108–119, 2019.

[109] ETSI, "Multi-access Edge Computing (MEC); Framework and Reference Architecture," *ETSI Group Specification 003*, 2019.

[110] 3GPP, "3GPP Technical Specification Group Services and System Aspects; System Architecture for the 5G System," 2019.

[111] D. Harutyunyan, N. Shahriar, R. Boutaba, and R. Riggio, "Latency and Mobility-Aware Service Function Chain Placement in 5G Networks," *IEEE Transactions on Mobile Computing*, pp. 1–1, 2020.

[112] M. Chowdhury, M. R. Rahman, and R. Boutaba, "ViNEYard: Virtual network embedding algorithms with coordinated node and link mapping," *IEEE/ACM Transactions on Networking*, vol. 20, no. 1, pp. 206 –219, 2012.

[113] F. Z. Yousaf, P. Loureiro, F. Zdarsky, T. Taleb, and M. Liebsch, "Cost Analysis of Initial Deployment Strategies for Virtualized Mobile Core Network Functions," *IEEE Communications Magazine*, vol. 53, no. 12, pp. 60–66, 2015.