Sorbonne University

Doctoral School of Informatics, Telecommunications and Electronics of Paris

EURECOM

Towards enforcing network slicing in 5G networks

Presented by Sihem BAKRI

Dissertation for Doctor of Philosophy in Information and Communication Engineering

Directed by Adlen KSENTINI

A Jury committee composed of:

Prof. Hacène Fouchal	University of Reims	Reviewer
MC-HDR. Daniel Négru	University of Bordeaux	Reviewer
Prof. Jérôme Härri	EURECOM	Examiner
Prof. Lynda Mokdad	University of Paris-Est	Examiner
MC. Joanna Moulierac	University of Nice	Examiner
Dr. Pantelis Frangoudis	TU Wien	Examiner

Publicly presented and defended on January 28th, 2021

Abstract

0.1 English

Mobile networks have experienced strong growth with the emergence of a new generation, namely 5G, where the demand for various services and applications becomes significant, and the required quality of service (QoS) becomes more critical. However, the current "one size fits all" of 4G network architectures cannot support these next-generation heterogeneous services criteria. Therefore, research around 5G aims to provide more adequate architectures and mechanisms to deal with this purpose. The 5G architecture is envisioned to accommodate the diverse and conflicting demands of services in terms of latency, bandwidth, and reliability, which cannot be sustained by the same network infrastructure.
Instead, network slicing provided by network virtualization allows the infrastructure to be divided into different slices. Each slice is tailored to meet specific service requirements allowing different services (such as automotive, Internet of Things, etc.) to be provided by different network slice instances. Each of these instances consists of a set of virtual network functions that run on the same infrastructure with specially adapted orchestration. Three main service classes of network slicing have been defined by the researchers as follows: Enhanced Mobile Broadband (eMBB), massive Machine Type Communication (mMTC), and ultra-Reliable and Low-Latency Communication (uRLLC).
One of the main challenges when it comes to deploying Network Slices is slicing the Radio Access Network (RAN). Indeed, managing RAN resources and sharing them among Network Slices is an increasingly difficult task, which needs to be properly designed. The goal is to improve network performance, and introduce flexibility and greater utilization of network resources by accurately and dynamically provisioning the activated network slices with the appropriate amounts of resources to meet their diverse requirements.
Our first contribution dealt with this latest problematic by proposing resource sharing algorithms running at the *Slice Orchestrator* (SO) level. The proposed algorithms compute the necessary radio resources to be used by each deployed network slice. These resources are adjusted periodically based on current estimates of achievable throughput performance derived from channel quality information, and in particular from the Channel Quality Indicator (CQI) values of the users of each

network slice retrieved from the RAN. CQI information is reported to base stations by the User Equipment (UE) following standard procedures, but extracting and frequently reporting it from base stations to the SO may result in significant communication overhead. To mitigate this overhead while maintaining at the SO level an accurate view of UE channel quality, we propose in our second and third contributions respectively: (i) a machine learning approach to infer the stability of UE channel conditions, and (ii) predictive schemes to reduce the CQI reporting intensity based on the inferred channel status.
On the other hand, network slicing enables the emergence of new players in the market: the Infrastructure, or slice, Provider (InfProv), which is the owner of the network infrastructure and may offer its resources as a service for a given cost, and the consumers, or tenants, requesting for a network slice from InfProv to get a target service with specific needs. However, as each provider has limited resources, it is challenging to have an optimal policy to decide which slice requests will be accepted (and/or rejected) by InfProv, and based on which criteria. On one hand, InfProv aims to maximize the network resources usage by accepting as many network slices as possible; on the other hand, the network resources are limited, and the network slices requirements regarding QoS need to be fulfilled. In this context, in the fourth contribution, we devise three admission control mechanisms based on Reinforcement Learning, namely Q-Learning, Deep Q-Learning, and Regret Matching. They allow deriving admission control decisions (policy) to be applied by InfProv to admit or reject network slice requests. We prove each mechanism's performance in terms of maximizing the InfProv while fulfilling the slice resources requirements.

0.2 Français

Les réseaux sans fil ont récemment connu une forte croissance, notamment la 5G, où la demande de différents services et applications augmente et la qualité de service (QoS) exigée devient plus importante. Toutefois, les architectures de réseaux sans fil actuelles, de type " une taille pour tous ", ne peuvent pas prendre en charge ces critères de services hétérogènes de nouvelle génération. Par conséquent, la recherche autour de la 5G vise à fournir des architectures et des mécanismes plus adéquats pour répondre à ce besoin. Plus précisément, l'architecture 5G est conçue pour répondre aux exigences variées et contradictoires des services, en termes de latence, de bande passante et de fiabilité, qui ne peuvent être assurées par la même infrastructure du réseau.
En effet, le découpage du réseau fourni par la virtualisation du réseau permet de diviser l'infrastructure en différentes tranches, chaque tranche est adaptée aux besoins spécifiques des services, où elle permet à différents services (comme l'automobile, l'Internet des objets...) d'être fournis par différentes instances de la tranche du réseau. Chacune de ces instances est constituée d'un ensemble de fonctions du réseau virtuel qui fonctionnent sur la même infrastructure avec une orchestration

spécialement adaptée. Les chercheurs ont défini trois grandes classes de services de découpage en réseau, qui sont: Enhanced Mobile Broadband (eMBB), massive Machine Type Communication (mMTC), and ultra-Reliable and Low-Latency Communication (uRLLC).
L'un des principaux défis du déploiement des tranches de réseau est le découpage du réseau d'accès radio (RAN). En effet, la gestion des ressources RAN et leur partage entre les tranches de réseau est une tâche particulièrement difficile, qui doit être bien conçue. L'objectif est d'améliorer les performances du réseau et d'introduire de la flexibilité et une plus grande utilisation des ressources du réseau en fournissant de manière précise et dynamique aux tranches de réseau activées les quantités de ressources appropriées pour répondre à leurs divers besoins.
Notre première contribution porte sur cette dernière problématique en proposant des algorithmes de partage de ressources fonctionnant au niveau du *Slice Orchestrator* (SO). Ces algorithmes calculent les ressources radio nécessaires à utiliser par chaque tranche de réseau déployée. Ces ressources sont ajustées périodiquement sur la base des estimations actuelles des performances de débit possibles obtenues. Ces performances de débit sont obtenues à partir des informations liées à la qualité du canal, et en particulier des valeurs de l'indicateur de qualité du canal (CQI) des utilisateurs de chaque tranche de réseau extraites du RAN. Les informations CQI sont transmises aux stations de base par l'équipement utilisateur (UE) selon des procédures standard. Cependant, leur extraction et leur transmission fréquente des stations de base au SO peuvent entraîner des frais généraux de communication importants. Pour atténuer ces frais généraux tout en maintenant une vue précise des qualités du canal de l'UE au niveau du SO, nous avons proposé dans nos deuxième et troisième contributions respectivement : (i) une approche d'apprentissage machine pour déduire la stabilité des conditions du canal UE, et (ii) un schéma prédictif pour réduire l'intensité de notification de CQI basée sur l'état du canal déduit.
De plus, le découpage du réseau permet l'apparition de nouveaux acteurs sur le marché : le fournisseur d'infrastructure, ou fournisseur de tranche (InfProv). L'infProv est le propriétaire de l'infrastructure de réseau et peut offrir ses ressources sous forme de service pour un coût donné. Les consommateurs, ou les locataires, qui demandent une tranche du réseau à InfProv pour obtenir un service particulier ayant des besoins spécifiques. Toutefois, comme chaque fournisseur dispose de ressources limitées, il est difficile d'avoir une politique optimale pour décider quelles demandes de tranche seront acceptées (et/ou rejetées) par InfProv, et sur la base de quels critères. D'une part, InfProv vise à maximiser l'utilisation des ressources du réseau en acceptant autant de tranches de réseau que possible ; d'autre part, les ressources du réseau sont limitées, et les exigences des tranches de réseau concernant la qualité de service (QoS) doivent être remplies. Dans ce contexte, dans la quatrième contribution, nous avons conçu trois mécanismes de contrôle d'admission basés sur l'apprentissage par renforcement, qui sont le Q-Learning, le Deep Q-Learning et le Regret Matching. Ces techniques permettant de dériver des décisions de contrôle

d'admission (politique) à appliquer par InfProv pour admettre ou rejeter les demandes de tranche de réseau. Finalement, nous avons prouvé la performance de chaque mécanisme en termes de maximisation de l'InfProv, tout en répondant aux besoins de tranches en terme de ressources.

Acknowledgements

This three-year journey has been an unforgettable experience for me. Going through ups and downs, with challenges every other day, I faced them with growing motivation and positivity.

I am happy I had the opportunity to live this experience, and I would like to thank IMT for the scholarship they offered.

I would like to express my sincere appreciation and gratitude to my supervisor Adlen KSENTINI for his excellent guidance, encouragement, and his availability. I thank him for all his useful suggestions, advice, insightful comments, and useful discussions throughout this journey.

Moreover, I sincerely thank the jury for accepting to be members of my defense jury.

It is my great pleasure to take this opportunity to thank all the people who helped me and accompanied me during my Ph.D. study. Good friends who have been there by my side when I needed them, thanks to all of them.

Finally, my heartfelt thanks go to my family. I thank my parents for their everlasting support.

Contents

0.1 English ii
0.2 Français iii

1 General Introduction **1**
1.1 Context 1
1.2 Contributions 3
1.3 Structure of thesis 5
1.4 Publications list 7

2 State of the Art **9**
2.1 4G Networks definition and architecture 9
2.1.1 Radio protocol architecture 10
2.1.2 Key technologies adopted in 4G 15
2.1.3 Why 5G? (from 4G to 5G) 16
2.2 5G Networks 17
2.2.1 5G technologies 18
2.3 Network slicing 21
2.3.1 Slicing the Core Network 24
2.3.2 Slicing the transport network 26
2.3.3 Slicing the Radio Access Network 28
2.3.4 Network slicing architecture: orchestration and management 33
2.4 5G network slicing challenges and open issues 35

3 Dynamic slicing of RAN resources for heterogeneous coexisting 5G services **38**
3.1 Motivations and state of the art 38
3.2 Architecture and assumptions 41
3.2.1 eMBB slice 42
3.2.2 uRLLC slice 42
3.3 A channel quality-driven algorithm for dynamic N_{pRB} estimation . . 45
3.4 Performance Evaluation 45
3.4.1 Scenarios and parameters 45
3.4.2 Results 48
3.5 Conclusion 51

4 Channel stability prediction to optimize signaling overhead in 5G networks using ML 52
4.1 Context and motivations . . . 52
4.2 Optimization of CQI signaling overhead state of the art . . . 54
4.2.1 Techniques based on frequency . . . 54
4.2.2 Techniques based on timing . . . 55
4.3 Channel stability prediction using machine learning . . . 55
4.3.1 Overview and objectives . . . 56
4.3.2 Steps to predict the channel's state . . . 57
4.4 Performance evaluation . . . 60
4.4.1 Test and validation phase evaluation . . . 60
4.4.2 Application phase evaluation . . . 61
4.4.3 Prediction quality evaluation . . . 62
4.5 Conclusion . . . 64

5 Data-Driven RAN Slicing Mechanisms for 5G and Beyond 65
5.1 Context and motivations . . . 65
5.2 RAN Slicing framework . . . 66
5.3 Algorithms for reducing the CQI reporting frequency . . . 67
5.3.1 Optimal Difference method . . . 68
5.3.2 Long Short-Term Memory method . . . 69
5.3.3 Comparison between methods . . . 71
5.4 Performance evaluation . . . 72
5.4.1 Model validation channel mobility/stability prediction with real data . . . 72
5.4.2 RAN slicing with optimization . . . 77
5.5 Conclusion . . . 79

6 On using reinforcement learning for network slice admission control in 5G: offline vs. online 80
6.1 Motivations and state of the art . . . 80
6.2 System model . . . 82
6.3 System Analysis . . . 84
6.3.1 Markov Decision Process model . . . 84
6.3.2 Admission control using QL . . . 86
6.3.3 Admission control using DQL . . . 88
6.3.4 Admission control using Regret Matching . . . 88
6.4 Performance evaluation . . . 89
6.5 Conclusion . . . 94

7 Conclusions and Perspectives 101
7.1 Conclusions . . . 101

7.1.1 Dynamic slicing of RAN resources for heterogeneous coexisting 5G services 101
7.1.2 Channel stability prediction to optimize signaling overhead in 5G networks using ML 102
7.1.3 Data-driven RAN slicing mechanisms for 5G and beyond . . 102
7.1.4 On using reinforcement learning for network slice admission control in 5G: offline vs. online 102
7.2 Perspectives 103

List of Figures

1.1 5G technology requirements [1]. 2
1.2 Thesis contributions organization 6

2.1 LTE network reference model [2] 10
2.2 RAN architecture in LTE [3] 12
2.3 Time-frequency resources in LTE [4] 14
2.4 CQI/ MCS mapping table in LTE [5] 15
2.5 The three layers in SDN architecture [6] 19
2.6 NFV architectural framework [7] 22
2.7 new core architecture [8] 25
2.8 gNB Logical architecture [9] 31
2.9 Functional split overview [10] 31
2.10 E2E network slicing architecture: high-level view [11] 33

3.1 Slices defined 39
3.2 2 leval MAC scheduler system model [12] 40
3.3 Latency vs. the number of uRLLC users. 47
3.4 Throughput vs. the number of uRLLC users. 48
3.5 Number of pRBs vs. the number of uRLLC users for a medium channel quality. 49
3.6 Number of pRBs vs. the number of uRLLC users for a good channel quality. 50

4.1 Our concept and methodology to reduce CQI monitoring overhead. 56
4.2 ML-driven channel stability prediction. 58
4.3 TPR of mobile channel for different CQI reporting frequencies. The x-axis represents the period between two consecutive CQI reports by a UE. 62
4.4 Prediction score for different CQI reporting frequencies. The x-axis represents the period between two consecutive CQI reports by a UE. 63

5.1 CQI reports exchange. 67
5.2 Steps to estimate the optimal number of CQI reports N_r^{opt}. 70
5.3 d_{pRB} error vs. N_r. 74

5.4 Optimal number of CQI reports (N_r^{opt}) over periods of $T = 200ms$ vs. mobility scenario. 75
5.5 d_{pRB} Error vs. mobility state using LSTM. 76
5.6 d_{pRB} normalized error vs. gain. 77
5.7 Throughput before and after applying the Optimal Difference method to reduce the CQI reporting frequency, for different user mobility scenarios. 78
5.8 Average throughput error across all considered forecasting period durations using LSTM, for different user mobility scenarios. 79

6.1 System model 82
6.2 $InfProv$ reward and penalty vs. $Time$ for slice arrival request rate= 2, and H_{time}= 5 & 20 tu 90
6.3 $InfProv$ reward and penalty vs. $Time$ for slice arrival request rate= 2, and H_{time}= 50 & 100 tu 91
6.4 $InfProv$ reward and penalty vs. $Time$ for slice arrival request rate=10, and H_{time}= 5 & 20 tu 95
6.5 $InfProv$ reward and penalty vs. $Time$ for slice arrival request rate=10, and H_{time}= 50 & 100 tu 96
6.6 Percentage of slice request reject vs H_{time} for slice arrival request rate= 10 97
6.7 Percentage of accepted slice type for slice arrival request rate=10, and H_{time}= 5 & 20 tu 98
6.8 Percentage of accepted slice type for slice arrival request rate=10, and H_{time}= 50 & 100 tu 99
6.9 Offline and Online DQL Average Reward of H_{time}= (5, 20, 50, 100) for slice arrival request rate= 10 100

List of Tables

2.1 LTE entities [2] 9
2.2 EPC entities 11

3.1 Simulation parameters 46

4.1 Accuracy and F1-score of NN and SVM algorithm 61

5.1 Mobility and stability results of the considered mobility scenarios using NN and SVM. 73

6.1 Number of resources: (i) available in InfProv, (ii) requested by each slice in UL and DL 92

Acronyms

3GPP ..... Third Generation Partnership Project

ADC ..... Application Distribution Controllers

AMC ..... Adaptive Modulation and Coding

AMF ..... Mobility Management Function

API ..... Application Programming Interface

BBU ..... BaseBand Unit

BSS ..... Business Support System

C-RAN ..... Cloud-RAN

CDR ..... Charging Data Record

CQI ..... Channel Quality Indicator

CN ..... Core Network

CSMF ..... Communication Service Management Function

CU ..... Central Unit

DQL ..... Deep Q Learning algorithm

DL ..... Downlink

DL-SCH ..... Downlink Shared Channel

DU ..... Distribution Unit

E2E ..... End to End

eMBB ..... enhanced Mobile Broadband slice type

eNB ..... evolved Node B, 4G RAN base station

EPC ..... Evolved Packet Core

EPS ..... Evolved Packet System

ETSI ..... European Telecommunications Standards Institute

FANS ..... Fixed Access Network Sharing

FDD ..... Frequency Division Duplex

FDM ..... Frequency-Division Multiplex

FDMA ..... Frequency-Division Multiple Access

gNB ..... g Node B, 5G RAN base station

HARQ ..... Hybrid Automatic Retransmission Request

HSS ..... Home Subscriber Server

ICN ..... Information-Centric Networking

InfProv ..... Infrastructure Provider

IoT ..... Internet of Things

IP ..... Internet Protocol

KPI ..... Key Performance Indicator

LSTM ..... Long and Short Term Memory

LTE ..... Long-Term Evolution

MAC ..... Medium Access Control Layer

MANO ..... Management and Orchestration

MCS ..... Modulation and Coding Scheme

MDP ..... Markov Decision Process

MIMO ..... Multiple-Input Multiple-Output

ML ..... Machine Learning

MME ..... Mobility Management Entity

MSE ..... Mean Squared Error

mMTC ..... massive Machine Type Communication slice type

MVNO ..... Mobile Virtual Network Operators

NBI ..... North Bound Interface

NFV ..... Network Function Virtualization

NFVO ..... NFV Orchestrator

NGFI ..... Next Generation Fronthaul Interface

NGMN ..... Next Generation Mobile Network

NN ..... Neural Network

NpRB ..... Number of physical Resource Block

NRF ..... NF Repository Function

NSMF ..... Network Slice Management Function

NSSMF ..... Slice Subnet Management Functions

OAI ..... Open Air Interface

OCS ..... Online Charging System

OFCS ..... Offline Charging System

OFDM ..... Orthogonal Frequency-Division Multiplexing

OSS ..... Operational Support System

PCF ..... Policy Control Function

PCRF ..... Policy and Charging Rules Function

PDCP ..... Packet Data Convergence Protocol

PDN ..... Packet Data Network

PDU ..... Protocol Data Unit

PGW ..... Packet GateWay

PHY ..... Physical layer

PMI ..... Precoding Matrix Indicator

pRB ..... physical Resource Block

QL ..... Q Learning algorithm

QoE ..... Quality-of-Experience

QoS ..... Quality-of-Service

QPSK ..... Quadrature Phase-Shift Keying

RAN ..... Radio Access Network

RB ..... Resource Block

RE ..... Reseource Element

RF ..... Radio Frequency

RI ..... Rank Indicator

RLC ..... Radio Link Control

RM ..... Regret Matching algorithm

RRC ..... Radio Resource Control

RRH ..... Remote Radio Headers

RRM ..... Radio Resource Management

SAC ..... Slice Admission Control

SC-FDE ..... Single-Carrier Frequency-Domain-Equalization

SDB ..... Slice-Dedicated Bandwidth

SDU ..... Service Data Unit

SDN ..... Software Defined Networking

SD-RAN ..... Software-Defined RAN

SGW ..... Service Gateway

SLA ..... Service Level Agreement

SMF ..... Session Management Function

SO ..... Slice Orchestrator

SPR ..... Subscription Profile Repository

SRM ..... Slice Resource Manager

SVM ..... Support Vector Machine

TBS ..... Transport Block Size

TDD ..... Time-Division Duplex

TNR ..... True Negative Rate

TPR ..... True Positive Rate

TTI ..... Transmission Time Interval

UE ..... User Equipment

UDM ..... Unified Data Management

UL ..... Uplink

UL-SCH ..... Uplink Shared Channel

UPF ..... User Plane Function

uRLLC ..... ultra Low-Latency Communication slice type

VAN ..... Virtual Access Node

VIM ..... Virtual Infrastructure Manager

VNF ..... Virtual Network Function

VMs ..... Virtual Machines

VoIP ..... Voice-over-IP

VPN ..... Virtual Private Network

vRB ..... virtual Resource Block

Chapter 1

General Introduction

1.1 Context

The evolution of wireless networks has been significantly revolutionized over the past decades, with the emergence of a new generation of wireless devices that are more powerful than the previous one. This evolution is mainly driven by the need for a better quality of experience (QoE), the continued increase in user wireless devices, and data usage [13]. In this context, the fourth generation (4G) Long Term Evolution (LTE) network has proven its effectiveness and is now part of everyday life. Indeed, 4G has certain advantages over the previous third-generation mobile network (3G), including higher bandwidth and data speeds, lower latency, higher network capacity, easier network integration through the IP network, and enhanced security. However, the increasing number of users, with the emergence of the Internet of Things (IoT), and new applications requiring very critical low latency and very high capacity, have complicated the management of network resources, mainly to accommodate a high number of demands, while supporting heterogeneous Quality of Service (QoS). This has caused a significant degree of complexity and problems in the network.

The new fifth generation of wireless communications (5G) aims to deal with the technical limits of 4G. It provides a significant improvement in the perceived QoS to users compared to the current 4G LTE network and, more importantly, opens new perspectives for the vertical industry [14].

5G focuses on several key perspectives, mainly:

- **Very low latency:** Latency is a very important metric in the definition of the 5G network. It is defined as the time it takes for data to transit back and forth between the 5G device and the access point. 5G aims to minimize latency up to 1 ms since it promises to support life-critical systems, services with a very low tolerance for delays, and real-time applications.
- **Ubiquitous connectivity:** Due to the continued exploding demand of data, the 5G network aims to meet the demand for capacity. It also seeks to allow

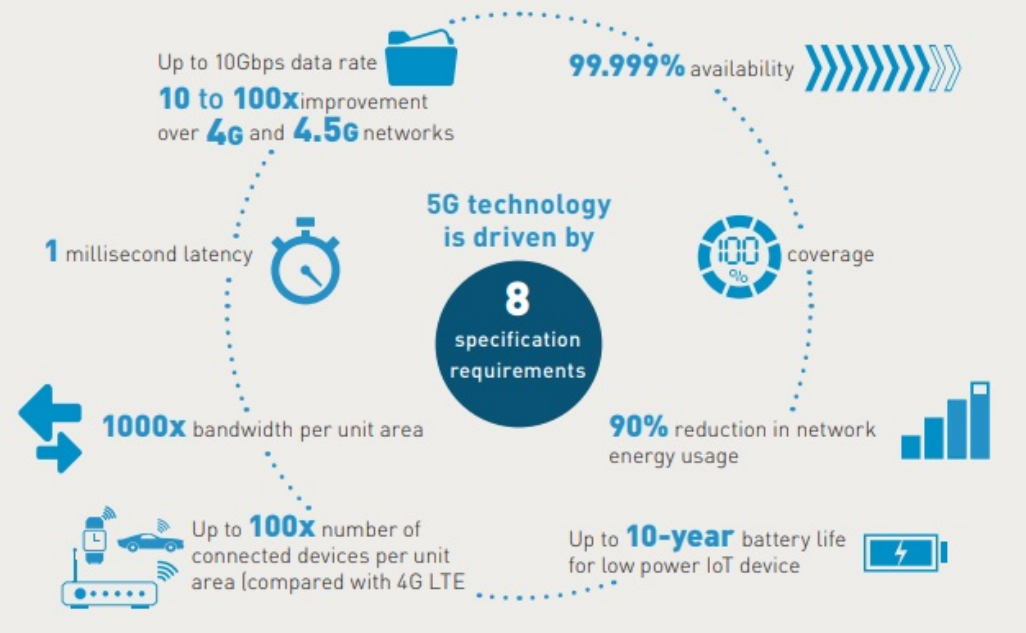


Figure 1.1: 5G technology requirements [1].

large numbers of device types to connect ubiquitously. Besides, it targets to provide uninterrupted user experience by achieving a user-centric vision through ubiquitous connectivity.

- **High-speed connection:** 5G will greatly improve the high-speed connection with fast data transmission and reception. It aims to achieve download speeds of one gigabit per second, and more than 10 times faster than what 4G provided.

Given that each 5G service may have different requirements in terms of latency, bandwidth, and reliability [15] as shown in Figure 1.1; the concept of one architecture fits all services of 4G cannot support these 5G heterogeneous demands. To this end, several research projects around 5G in the academic and industrial areas have proposed new architectures and technologies adapted to the 5G requirements. The latest 5G technologies are based on: (i) network softwarization and virtualization and (ii) network slicing. The network virtualization relies on the concept of architecture, design, deployment, and management of network components, mainly based on the programmability properties of software [16]. It enables adaptability, flexibility, and total reconfiguration of a network, by reducing the cost and optimizing the process of the global maintenance of the network life cycle. On the other hand, network slicing is an end-to-end logical network with a set of isolated virtual resources. It is intended to ensure isolation, customization of services, and support of several tenants on a common physical network infrastructure while relying on a physical and logical separation of network resources. The key idea of network slicing is to "slice" the original network architecture into separate logical and in-

dependent networks configured to efficiently address the different service demands. Three main slice types are defined in 5G [12] [16] as follows:

- enhanced Mobile Broadband (eMBB), which requires a high data rate, even in dense areas, for instance: 3D video, immersive Gaming, and AR/VR.
- massive Machine Type Communication (mMTC) to support IoT services citing smart sensors, smart cities, and massive IoT.
- ultra-Reliable and Low-Latency Communication (uRLLC) to support services requiring low-latency access like self-driving cars, augmented reality, and industry 4.0.

It is worth noting that, network slicing includes slicing the 5G radio access network (RAN), 5G core network (CN), and 5G transport network. It relies on the concept of network virtualization and Softwarization (Software Defined Networking - SDN and Network Functions Virtualization - NFV) to share a common infrastructure. These concepts also enable it (i.e., network slicing) to build virtual instances (slices) of the network tailored to the different 5G services' needs.

Although it is designed to provide significant improvements and solutions to support the new 5G network requirements, network slicing introduces critical research challenges and policy directions [17]. The main network slicing challenges include: sharing and the algorithmic aspects of resource allocation, mobility management, dynamic creation and management, network slice isolation and security, and virtualization of wireless resources [17].

For this purpose, this thesis contributes a study that touches upon various critical issues in network slicing. In particular, we have investigated the challenges related with resource sharing and resource allocation, with a special focus on the RAN domain

1.2 Contributions

This Ph.D. thesis is divided into two main parts: (i) the first part investigates the improvement of network performance by addressing the slice resource sharing issue; (ii) the second part focuses on determining a trade-off between the bandwidth limit and the network slicing revenues in the network slicing resource sharing issue. The contributions of this thesis are summarized in Figure 1.2 and described in detail as follows:

1. One of the main challenges when it comes to deploying network slices is slicing the RAN. Indeed, managing the limited RAN resources and sharing them among network slices is an increasingly difficult task, which needs to be properly designed. The goal is to improve network performance and introduce

flexibility and greater utilization of network resources. For this reason, it is necessary to accurately and dynamically provision the activated network slices with the appropriate amounts of resources to meet their diverse requirements. Therefore, each slice requires different resources for a determined period of time, according to several constraints such as the number of users hosted by the slice, the required slice application's throughput, and the latency. For instance, eMBB slices request high bandwidth, while uRLLC slices aim to minimize latency and maximize reliability.

To this end, in this thesis, we first propose algorithms that allow to dynamically define the amount of resources to be allocated to each admitted network slice. The proposed algorithms consider each slice's specific requirements and the varying conditions of the radio environment while considering the bandwidth limit. It is worth noting that this algorithm runs at the slice orchestrator (SO) level, which is responsible for cross-slice resource sharing. In this contribution, the considered algorithms to drive the number of resources are based on the Channel Quality Indicator (CQI) values reported to the SO by the base station (i.e., evolved NodeB (eNB)). In fact, the CQI is a value periodically reported by UEs to eNB, then transmitted from the eNB to the SO, where it conveys the current communication channel quality of each UE. In our solution, the SO periodically collects the CQI values from the eNB to update the calculation of the amount of the resource needed by each slice. Nevertheless, the periodic transmission of CQI information between eNB and SO incurs signaling overhead that needs to be mitigated.

The second contribution fits this gap and focuses on CQI reports periodically sent from the eNB to the SO. Thus, we propose mechanisms to optimize the reporting process with the aim of reducing signaling overhead between the eNB and the SO. This latest mechanism is based on machine learning (ML) algorithms that predict UE channel stability to decide if the CQI is necessary to be reported or not.

In the third contribution, we design predictive schemes to reduce the CQI reporting intensity based on the inferred channel status and tune the reporting frequency appropriately to reduce the CQI monitoring overhead. Our contribution here consists of reducing the signaling overhead between the SO and the eNB, by optimizing the reporting of CQI information via limiting the amount of unnecessary transmitted messages while ensuring that SO has an accurate view of the channel conditions at the eNB. Ideally, the eNB should notify the SO only when the UE channel condition has changed. In order to verify the efficiency of this proposed mechanism, we have integrated it with the resource allocation algorithms. This study allows us to verify whether resource allocation algorithms are properly driving the number of resources required by each slice type, despite reducing CQI reports.

2. In addition to the significant performance improvements, network slicing opens the market to new players, mainly: (i) the infrastructure provider (InfProv), which is the owner of the infrastructure that provides to the tenants network slices corresponding to a certain fraction of network resources; (ii) the tenants, which can acquire a network slice from the InfProv to provide a specific service to their customers. Therefore, achieving a fair usage of network resources is of vital importance in Slice-ready 5G network. The dilemma of which network slice has to be accepted or rejected is very challenging for the InfProv. On one hand, the InfProv aims to maximize the network resource usage, by accepting as many network slice as possible; on the other hand, network resources are limited, and the requirements of network slices regarding QoS need to be fulfilled.

   In the first contribution, we have shown that our proposed algorithms could estimate the appropriate amount of resources to allocate to each slice type. However, this is no more respected after exceeding the network resources limit. To overcome this drawback, in the fourth contribution, we deal with the bandwidth, i.e., network resources limit while considering the concept of revenues in the slice resource allocation. Therefore, we devise three admission control mechanisms based on Reinforcement Learning, namely Q-Learning, Deep Q-Learning, and Regret Matching, which allow deriving admission control decisions (policy) to be applied by InfProv to admit or reject network slice requests. Our solution enables maximizing InfProv's revenues and respecting the slice QoS by providing all the resources necessary by each network slice request within the network resources limits during the slice lifetime.

1.3 Structure of thesis

The content of the following chapters is summarized in this section for the ease of the reader.

- Chapter II introduces state of the art on the 5G network in general and on network slicing in particular, including relevant related works. In this chapter, we start by giving a general overview of 4G network, where we present the 4G architecture and the technologies adopted, and the 4G limits. We detail then some 5G architectures proposed by the research community and the main challenges involved with it. We also present an appropriate study on the new technologies used to successfully deploy 5G network and network slicing, such as SDN and NFV. Afterward, we present a detailed survey of the end-to-end network slicing, including the CN, RAN, and transport network. Finally, we give the main 5G network slicing challenges and open issues.

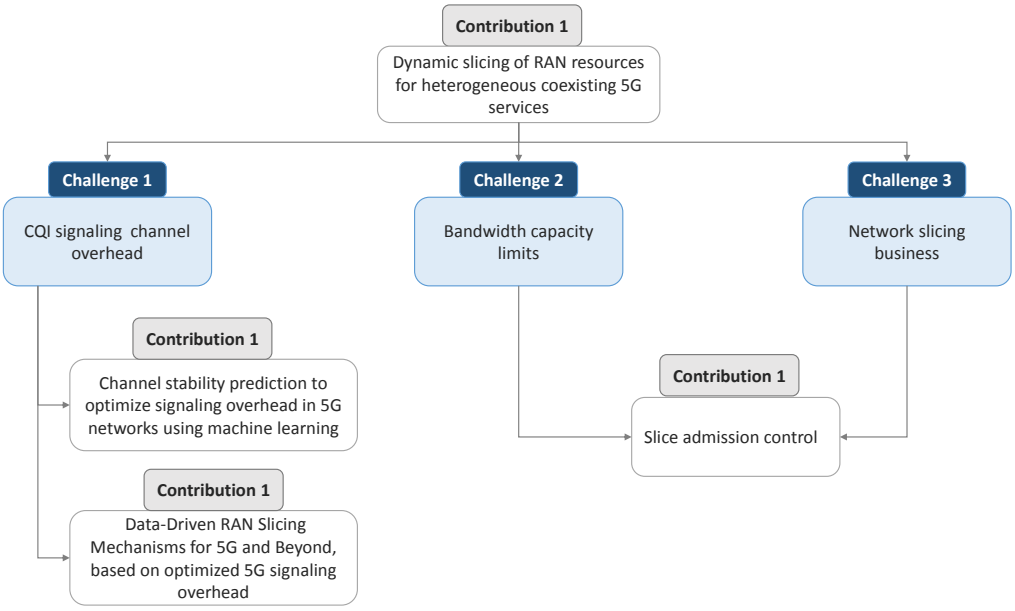


Figure 1.2: Thesis contributions organization

- Chapter III presents a resource sharing algorithm that aims to compute and dynamically adjust the necessary radio resources to be used by each deployed network slice [18]. Therefore, we first propose an algorithm that derives the radio resources, namely pRBs, needed by each slice, according to its requirements in terms of latency and throughput. Then we design a RAN-aware dynamic slicing algorithm. This algorithm exploits per-user RAN-level information to accurately translate the derived service rate to an appropriate pRB assignment, using the CQI reports periodically obtained from eNBs.

- In chapter IV, we present a model that aims to detect whether the channel is static or mobile over time. On one hand, if the channel is static (i.e., its conditions are mostly stable or low-mobility), the reported CQI values between the eNB and SO remain constant or show minimal variation, which does not impact radio resource allocation. On the other hand, if the channel is mobile (i.e., it varies significantly), the CQI values exhibit significant fluctuations. Consequently, it is crucial that the SO is informed about the changed channel quality information. Indeed, this information allows it (i.e., the SO) to determine and update the appropriate amount of resources to be allocated for the different slices. Therefore, we use different ML algorithms, namely Support Vector Machine (SVM) and Neural Network (NN) in order to classify whether the CQI values will tend to change or not (i.e., stable vs. unstable channel.) [19].

- Chapter V details the development of a mechanism to meet network slices' requirements while reducing the number of CQI signaling messages between

the eNB and the SO. This contribution aims to limit the amount of unnecessary transmitted CQI messages. Based on the channel mobility identification previously proposed, we have used Long Short Term Memory (LSTM), a ML algorithm, and another proposed method based on minimizing errors between predicted CQI and collected CQI to reduce CQI reporting frequencies. Subsequently, we verify that this reduction in CQI reporting does not affect the slice requirements, such as the throughput for the eMBB slice.

- Chapter VI presents the development of a slice admission control algorithms to be run at the InfProv level aiming at deriving an optimal policy to decide if an arrived network slice request has to be accepted or rejected. Indeed, it is difficult to find the optimal policy that, on one hand, increases the revenue of the InfProv and allows an optimal usage of the infrastructure; on the other hand, guarantees the requirement of the admitted network slice in terms of QoS to avoid violating the Service Level Agreement (SLA). This contribution aims to satisfy each slice's needs in terms of the number of resources requested while maximizing the revenues of the InfProv. To this end, we model this problematic using the Markov Decision Process (MDP) and solve it using reinforcement learning algorithms, aiming to seek the optimal policy to increase the InfProv revenue while reducing the penalty to pay due to SLA violation. Three algorithms are introduced: Q-Learning (QL), Deep Q-Learning (DQL), and Regret Matching (RM). Besides deriving the optimal policy, we shed light on the proposed algorithms' ability to run offline or online, which is a crucial criterion. Indeed, offline solutions require a training phase before being used, which is sometimes costly, but they generally achieve the best results, while online solutions are trained on the fly using only observable information of the controlled system.

- Finally, Chapter VII concludes this thesis and presents different perspectives and future research directions.

1.4 Publications list

1) **Sihem BAKRI**, Pantelis Frangoudis and Adlen Ksentini: "Dynamic slicing of RAN resources for heterogeneous coexisting 5G services,"in *IEEE Global Communications Conference (GLOBECOM)*, Waikoloa Hawaii, United States 2019.

2) **Sihem BAKRI**, Maha Bouaziz, Pantelis Frangoudis and Adlen Ksentini: "Channel stability prediction to optimize signaling overhead in 5G networks using machine learning," in *Proc. IEEE International Conference on Communications (ICC)*, Dublin Ireland, 2020.

3) **Sihem BAKRI**, Pantelis Frangoudis, Adlen Ksentini, and Maha Bouaziz: "Data-driven RAN slicing mechanisms for 5G and beyond", submitted to IEEE Transactions on Network and Service Management.

4) **Sihem BAKRI**, Bouziane Brik and Adlen Ksentini: "On using reinforcement learning for network slice admission control in 5G: offline vs. online", Wiley International Journal of Communication Systems, Status: accepted.

Chapter 2

State of the Art

In this chapter, we will give an overview of the 4G LTE wireless network and introduce 5G and its related technologies, mainly network softwarization and network slicing. In particular, we will detail the concept of network slicing. We will conclude by discussing the challenges of network slicing, already defined in the literature.

2.1 4G Networks definition and architecture

The 4G or LTE, also known as EPS (Evolved Packet System), is the fourth generation of mobile telecommunications technology. This technology is introduced by 3GPP in order to improve network performance by defining an End-to-End (E2E) all-IP architecture for the core and radio access networks [2]. Note that the E2E all IP network means that all traffic flows are transferred based on IP protocol.This last functionality of providing mobile Internet access, represents the main characteristic of 4G compared to previous technologies.

Table 2.1: LTE entities [2]

Entity	**Description**
UE	UE connect to an eNB using LTE-Uu interface
eNB	It is the hardware that is connected to the UE to provide radio interfaces and performs Radio Resource Management (RRM) functions, such as scheduler, eNB measurement configuration, connection mobility control, radio admission control, inter cell interference coordination and radio bearer control

The LTE architecture is mainly composed of two parts: (i) Evolved Packet Core (EPC) part which deals with the technology related to the CN, and (ii) RAN

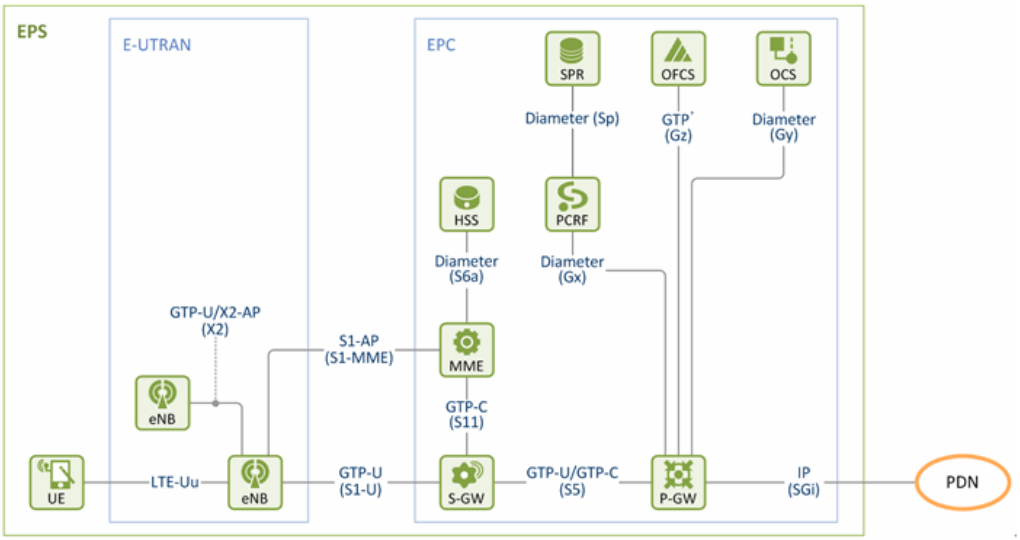


Figure 2.1: LTE network reference model [2]

which deals with the technology related to a radio access network (E-UTRAN). In this context, Figure 2.1 shows an LTE network reference model, composed of RAN entities (eNB and UE), EPC entities (Packet Gateway (PGW), Service Gateway (SGW), Mobility Management Entity (MME), Policy and Charging Rules Function (PCRF), Subscription Profile Repository (SPR), Home Subscriber Server (HSS), Offline Charging System (OfCS) and Online Charging System (OCS)), and the interfaces between these entities. Besides, Tables 2.1 and 2.2 describe and detail the role of each RAN and EPC entity respectively. A Packet Data Network (PDN) presents an internal or external IP domain of an operator that a UE communicates with and provides the UE with services, such as IP Multimedia Subsystem or the Internet [2].

2.1.1 Radio protocol architecture

To better understand the RAN architecture, we illustrate the RAN protocol architecture, including the control plane and user plane in Figure 2.2. Note that the MME is a CN entity. The different protocol entities of the RAN shown in Figure 2.2 are described as follows:

A. Packet Data Convergence Protocol:

Packet Data Convergence Protocol (PDCP) compresses IP headers (i.e., information at the beginning of an IP packet) to reduce the number of bits to be transmitted over the radio interface. PDCP is also responsible for the control plane, integrity protection of the transmitted data, ciphering, in-sequence transmission, and duplicate removal for handover. The PDCP protocol executes the relevant decryption and decompression operations at the receiver side [3].

Table 2.2: EPC entities

Entity	Description
MME	The main functions provided by the MME are as follows: • It manages the signalling (control plane) between the terminals (UE) and the LTE CN. • It dialogues with the HSS in order to access and store terminal profiles and security data. • Selects the PGW and SGW to be used later for data transfers when a given terminal connects to the network. • EPS bearer management.
SGW	It collects the data to be sent to the PGW from the UEs via the base stations. It also participates in the transmission of data in the opposite direction, from the PGW to the UEs.
PGW	It routes Internet data to the UE and vice versa; since it is a gateway between the operator's IP network and the Internet, it also provides some security functions.
HSS	The HSS is the central data base that stores the user profiles, and it provides user authentication information and user profiles to the MME.
PCRF	PCRF is a node which determines the policy rules for charging and bandwidth in a multimedia network.
SPR	A SPR is a logical entity providing subscription information to the PCRF. Receiving the information, the PCRF performs subscriber-based policy and creates a PCC rules.
OCS	OCS is a system allowing a communications service provider to charge function and control, in real time, based on service usage.
OFCS	OFCS provides Charging Data Record (CDR) based charging information.

B. Radio-Link Control:

Radio-Link Control (RLC) is responsible for retransmission handling, concatenation/segmentation, in-sequence delivery to higher layers, and duplicate detection. The RLC delivers services to the PDCP in the form of radio bearers [3].

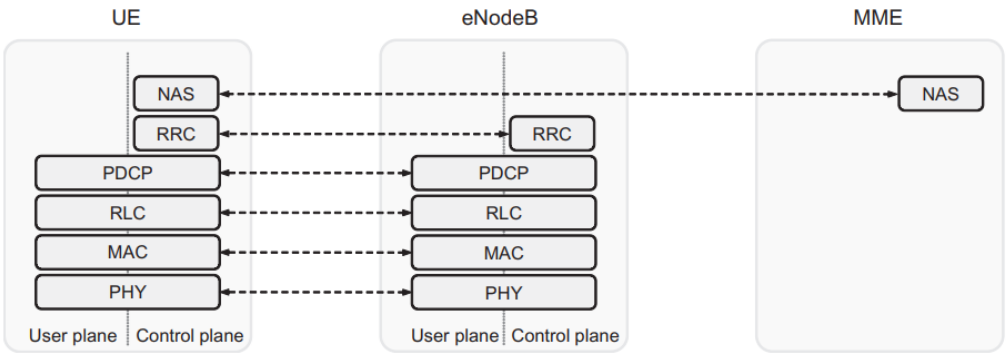


Figure 2.2: RAN architecture in LTE [3]

C. Medium-Access Control:

Medium-Access Control (MAC) manages Hybrid Automatic Retransmission Request (HARQ) retransmissions, uplink (UL) and downlink (DL) scheduling and multiplexing of logical channels. The HARQ protocol part is present in both the receiving and transmitting ends of the MAC protocol. The scheduling functionality is located in the eNB for both DL and UL. The MAC offers services to the RLC in the form of logical channels [3]. In this context, the logical and transport channels related to the MAC, and the scheduling functionality of the MAC layer, are described as follows:

- **Logical and Transport Channels:**

  The RLC receives services from the MAC in the form of logical channels. A logical channel is defined by the type of information it transports. It is considered as a control channel. It allows the transmission of control and configuration information concerning the operation of an LTE system. It may also be considered as a traffic channel used for user data.

  The MAC layer uses services from the physical layer (PHY) in the form of transport channels. Data on a transport channel are organized into transport blocks. In the absence of spatial multiplexing, in each Transmission Time Interval (TTI), at most, one transport block of dynamic size is transmitted over the radio interface to/from a terminal. In the presence of spatial multiplexing i.e., Multiple-Input Multiple-Output (MIMO), there can be up to two transport blocks per TTI.

- **Scheduling:**

  Shared-channel transmission is one of the basic principles of LTE RAN. It consists in dynamically sharing time-frequency resources between users. The scheduler is part of the MAC layer controlling the resource allocation (in terms of so-called RBs) pairs of UL and DL. The RBs represent a time-frequency unit of 1 ms time and 180 kHz frequency. The scheduler allows dynamic

scheduling between eNB and UEs. The eNB makes a scheduling decision every 1 ms interval and sends scheduling information to all selected UEs. A semi-static scheduling pattern maybe signaled in advance, .i.e. semi-persistent scheduling in order to reduce the control-signaling overhead.

In LTE, the DL and UL are separated. The DL scheduler controls which UEs to transmit to, and the set for each of these UEs the RBs upon which the UE's DL channel should be transmitted. The UL scheduler controls which UEs are to transmit on their respective UL channel and on which UL time–frequency resources (including component carrier).

The main aim of a scheduler is to take advantage of channel variations between UEs and to schedule resource transmissions to the UE having advantageous channel conditions. In this context, using Orthogonal Frequency-Division Multiplexing (OFDM), which allows the control of channel variations in the frequency and time domains through channel-dependent scheduling, presents an advantage in LTE. DL channel-dependent scheduling is provided by the channel-state reports transmitted by the UE. The channel-state reports reflect the instantaneous channel quality in the frequency and time domains. For the UL channel-dependent scheduling, the channel state information can be derived from a sounding reference signal transmitted from each UE for which the eNB desires to rate the UL channel quality. To this end, the UE transmits buffer-status information to the eNB using a MAC message.

D. Physical Layer:

PHY Layer manages multi-antenna mapping, modulation/demodulation, coding/decoding, and other typical functions in the PHY. The PHY provides services to the MAC layer in the form of transport channels. Data transmission in DL and UL uses the transport-channel of types DL Shared Channel (DL-SCH) and UL Shared Channel (UL-SCH) respectively. In the case of spatial multiplexing, there is at most one or two transport blocks per TTI on a DL-SCH or UL-SCH. There is one DL-SCH (or UL-SCH) per component carrier, in the case of carrier aggregation [3]. At the transmitting side, each layer outputs a Protocol Data Unit (PDU) to the layer below and receives a Service Data Unit (SDU) from a higher layer, for which the layer provides a service.
The transmission resources and control signaling in the PHY layer are described as follows:

- **Physical Transmission Resources**

  The basic transmission scheme used for UL and DL in LTE is OFDM. The subcarrier spacing of LTE OFDM for both UL and DL is 15 kHz. In the time domain, LTE transmissions are organized into radio frames of length 10 ms. Each frame is divided into ten subframes equally sized of length 1 ms. Each

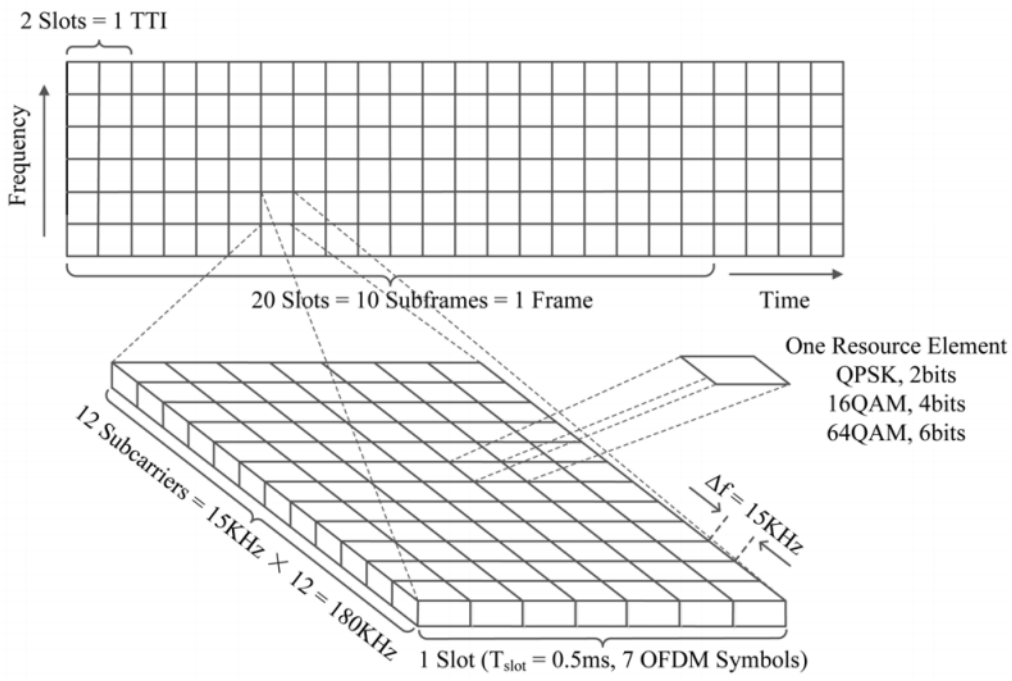


Figure 2.3: Time-frequency resources in LTE [4]

sub-frame is composed of two slots of equal size and length $T_{slot} = 0.5$ms, where each slot contains a number of OFDM symbols.

A resource element is the smallest physical resource in LTE, consisting of a subcarrier during one OFDM symbol. The resource elements are grouped into RBs. A RB is composed of one 0.5 ms slot in the time domain and 12 consecutive subcarriers in the frequency domain. To illustrate these details, Figure 2.3 shows a presentation of the time-frequency resources in LTE. The PHY layer characteristics in LTE enable a carrier to consist of many RBs in the frequency domain, ranging between six RBs minimum to 110 RBs maximum. This translates into a transmission bandwidth ranging from 1 MHz to 20 MHz. The above definition of RB applies to both UL and DL transmission directions.

- **PHY layer control signaling:**

  Control signaling instructions are provided to support the transmission of UL and DL transport channels. The DL and UL control signaling is information partly originating from PHY layer, and partly from MAC layer. The DL control signaling is composed of scheduling information, allowing the UE to properly receive, demodulate, and decode the DL-SCH on a component carrier. It is also used to provide other transmission services, including informing the UEs about the transport format and resources for UL transmission.

  The UL control signaling consists of: (i) channel-state reports related to the

CQI	Modulation	Bits/Symbol	REs/PRB	N_RB	MCS	TBS	Code Rate
1	QPSK	2	138	20	0	536	0.101449
2	QPSK	2	138	20	0	536	0.101449
3	QPSK	2	138	20	2	872	0.162319
4	QPSK	2	138	20	5	1736	0.318841
5	QPSK	2	138	20	7	2417	0.442210
6	QPSK	2	138	20	9	3112	0.568116
7	16QAM	4	138	20	12	4008	0.365217
8	16QAM	4	138	20	14	5160	0.469565
9	16QAM	4	138	20	16	6200	0.563768
10	64QAM	6	138	20	20	7992	0.484058
11	64QAM	6	138	20	23	9912	0.600000
12	64QAM	6	138	20	25	11448	0.692754
13	64QAM	6	138	20	27	12576	0.760870
14	64QAM	6	138	20	28	14688	0.888406
15	64QAM	6	138	20	28	14688	0.888406

Figure 2.4: CQI/ MCS mapping table in LTE [5]

conditions of DL channel, used to support DL scheduling; (ii) scheduling requests, stating that a terminal requires UL resources for UL-SCH transmissions; and (iii) HARQ acknowledgments for the DL-SCH transport blocks received.

It is worth noting that the channel state reports contain Channel-Quality Indicator (CQI), Rank Indicator (RI), and Precoding Matrix Indicator (PMI).

The CQI is the information that UE sends to the network (eNB) carrying: (i) how bad or good the communication channel quality is, and (ii) indicates the transport block size of the UE's needed data, which in turn can be directly converted into throughput. CQI information is reflected in a value between 0 and 15 (4 bits) and mapping between CQI and modulation and coding scheme (MCS), transport block size. In this context, each CQI value has a number of RB (N_{pRB}) and MCS, to properly allocate the resources for each of UEs. Table 2.4 contains a range of MCS that can be used for each CQI index. In addition, it contains the code rate between the MCS and the N_{pRB}, the Transport Block Size (TBS), Modulation, Bits/Symbol, Resource elements per physical RB (REs/PRB), and N_{pRB}.

2.1.2 Key technologies adopted in 4G

The following key features can be observed in all suggested 4G technologies:

- IP-based femtocells that represent home nodes connected to fixed Internet broadband infrastructure. This technology offers increased capacity and coverage in both home and office environments.

- PHY layer transmission techniques [3] are as follows:

  - MIMO: to attain ultra-high spectral efficiency using spatial processing including multi-antenna and multi-user MIMO.
  - Frequency-domain-equalization, for example, Single-Carrier Frequency-Domain-Equalization (SC-FDE) in the UL or multi-carrier modulation (MC-OFDM) in the DL, to exploit the frequency selective channel property without complex equalization.
  - Frequency-domain statistical multiplexing, for example, OFDMA in DL or (single-carrier FDMA) in the UL: variable bit rate through the assignment of different sub-channels to different users according to channel conditions.
  - Turbo principle error-correcting codes: to minimize the required SNR at the reception side

- Link adaptation: adaptive modulation and error-correcting codes, which enables fast responses to changing nature of transmission channel [3].

- Channel-dependent scheduling: As defined in section 2.1.1-C, the scheduling technique aims to use the time-varying channel. It represents a crucial challenge in the 4G networks and the next generation, such as 5G. In this context, the RRM block exploits a mix of advanced MAC and Physical functions, like CQI reporting, link adaptation through Adaptive MCS, resource sharing, and HARQ. Therefore, the conception of effective resource allocation policies becomes crucial. The efficient use of radio resources is a key factor in achieving system performance objectives and satisfying user needs according to specific QoS requirements [3]. The packet scheduler is in charge of assigning portions of the spectrum shared between users and maximizing the spectral efficiency. This is achieved through a specific resource allocation policies that reduce or make negligible the impact of channel quality drops. This packet scheduler works at the eNB. Knowing that the channel quality is affected by high variability in time and frequency domains on wireless links due to several causes, such as fading effects, Doppler effect, and multipath propagation, the OFDMA systems are adopted as a channel-aware solution. In fact, these solutiopns make it possible to exploit channel quality variations by assigning higher priority to users experiencing better channel conditions.

2.1.3 Why 5G? (from 4G to 5G)

It is worth noting that the 4G technology had enabled the delivery of high-quality use cases, including video and calling on the go, live TV, which was not possible using third-generation mobile networks (3G). However, the increase in the number of users with heterogeneous requirements has led to increased network congestion. Therefore, 4G is reaching the technical limits of how much data it can transfer

quickly over the spectrum blocks.
5G mobile networks [13] are designed to provide new and enhanced services that make life easier in several areas introduced in the Internet of Things (IoT), such as smart cities, health care, agriculture, transportation, and manufacturing. To accommodate these services, 5G has to meet critical requirements in terms of low latency, high reliability, high bandwidth, and the support for massive numbers of connected devices. In this context, 5G requires deploying many more devices in a reliable, secure, and uninterrupted manner in the same area, where the 4G architecture and its technologies have not been able to achieve. Overall, due to the new technologies, spectrum, and frequencies it uses, 5G has several advantages over 4G, citing: lower latency, higher speeds, capacity for more connected devices, less interference and higher efficiency, etc.

2.2 5G Networks

The motivation for migrating networking systems from traditional wireless networks to the Next Generation Networks 5G was developed based on the benefits of reduced backbone costs, controllable QoS, the possibility of fast and new service deployment, compatibility between wireless and fixed networks, centralized network management, etc. In this context, multimedia applications based on existing network services, such as data, voice, and high-speed video transmission, will be offered as an important outcome of the new generation deployment. In fact, the integration topology of fixed and mobile services allows providing a low-cost service at high data rates. Therefore, 5G aims to achieve the best performance in terms of data speeds of 1 Gbps, low latency, coverage capability, energy consumption, and better security and energy efficiency over spectral compared to previous networking systems [20] [21] [22].
Research on 5G has been initiated by many organizations, projects, and standardization forums and has not been precisely defined and characterized. Such research on 5G might be driven by the main limitations of the previous technologies, mainly:

- Provide wireless communication with no limitation of coverage edge, density zone, and access policy.
- Support for high-resolution multimedia (HD) broadcasting service.
- Ensure faster data speeds than the previous generations.
- Support new services based on wearable devices.
- Cover and manage massive inter-device connections (connection of Things).

2.2.1 5G technologies

The previous network infrastructure lacks the functionality needed in the new generation, making it unsuitable for covering the 5G network requirements. For instance, the traditional infrastructure's network management is very complex since the changes in network configurations are manual. Therefore, in order to address the limitations of the current networks, SDN and NFV present an emerging technology. SDN and NFV allow the separation of network control from the underlying data planes or switching devices and virtualization of entire classes of network functions, respectively [20]. SDN offers flexibility in changing network policies, easy hardware implementation, and facilitates network innovation and evolution [23] [24]. Thanks to its architecture that breaks the virtual integration between the data plane and the control plane using a centralized SDN controller. The integration of SDN with NFV provides a global view of the entire network using an open interface such as OpenFlow [25] and the centralized network controller. SDN can support new programs and services at any level of user requirement or need. The SDN and NFV have attracted considerable interest from academies and enterprises in recent years. They represent an essential step in the evolution and development of future network infrastructures.
In the following subsections, we will detail the virtualization technologies mentioned above, citing SDN and NFV.

A. SDN

Knowing that traditional networks' static architecture is decentralized and complex, SDN is designed to provide flexibility and easy troubleshooting to these existing networks. SDN technology is an emerging architecture to network management that enables efficient network programming and dynamic configuration. It aims to improve monitoring and network performance and bring traditional network management closer to cloud computing. SDN separates network packets' forwarding process (data plane) from the routing process (control plane) to centralize network intelligence in one network component. The control plane consists of one or more controllers. They are considered as the brain of the SDN network where all intelligence is embedded. However, the intelligent centralization has its own drawbacks when it comes to scalability, security, and elasticity [26] which present the main issue of SDN.
SDN is commonly associated with the OpenFlow protocol. It aims to describe a controller that communicates with network devices such as switches and routers by providing a standardized traffic management method. The devices supporting OpenFlow consist of two logical components: (i) an exposed OpenFlow application programming interface (API) that handles the exchanges between switch/router and controller, and (ii) a flow table that defines how to process and forward packets within the network. In this context, Figure 2.5 shows the reference SDN architecture

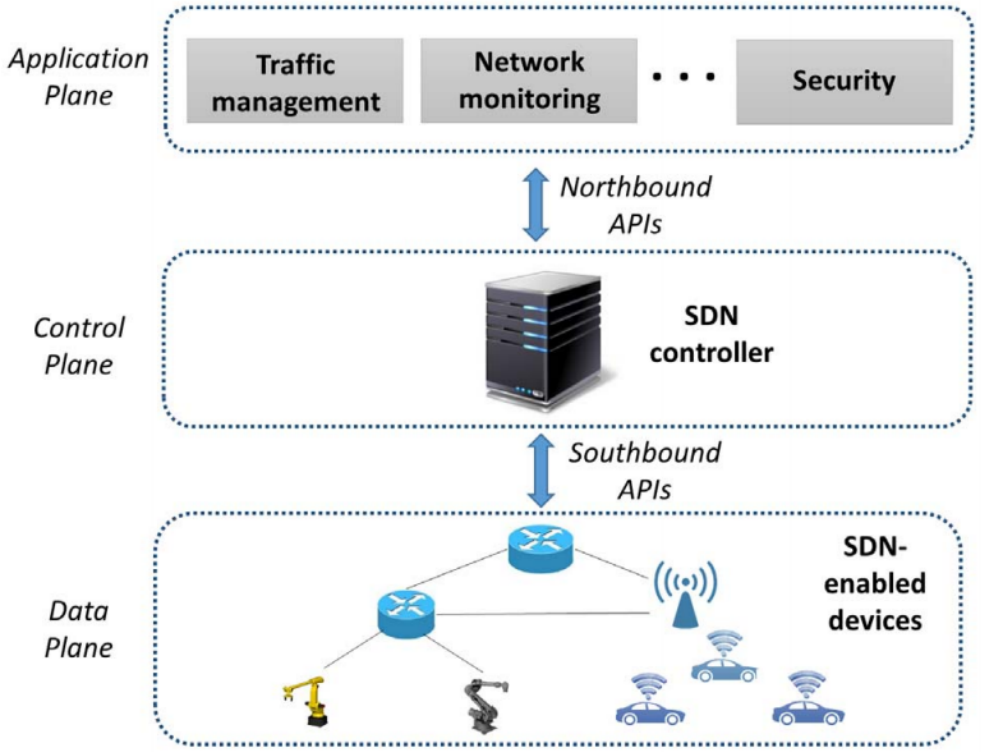


Figure 2.5: The three layers in SDN architecture [6]

model [27] composed of three layers, as follows:

- The SDN applications can specify their requirements for traffic management in the underlying networks through Northbound APIs.
- The SDN controller, which is responsible for the control plane, bridges the application plane and the data plane. It translates the requirements of the applications into appropriate forwarding rules to be enforced by the underlying network switches. The southbound API allows the SDN controller to access functions provided by the SDN-enabled switching devices. These functions may include managing packet forwarding rules and reporting network status.
- The data plane covers network elements (e.g., routers and switches) used to collect network status information and process packets based on rules provided by the SDN controller, such as traffic statistics network topology.

B. NFV

NFV technology was initially created by various major service providers to transition from a hardware-oriented to a software-oriented infrastructure. Software features accelerate the deployment of new network services and drive revenue growth

while reducing operational costs. Basically, NFV technology is a new type of infrastructure to manipulate [28]. It aims to virtualize network functions such as Application Distribution Controllers (ADCs), Firewell Web application transcoders, load balancers, etc. Virtualization aims to provide a flexible software design. Therefore, NFV enables additional dynamic schemes to create and manage network functions since the existing networking services are supported by diverse network functions that are connected in a static way [29].
The key concept of NFV is the virtual network function (or VNF). VNF is used to manage specific network functions that run on one or more virtual machines (VMs) on the hardware network infrastructure (routers, switches, etc.). These VNFs can be connected or combined as components to provide an optimal network communication service [28]. They are designed to consolidate and provide the network components necessary to support a completely virtual environment. A set of VNFs can be composed together in order to reduce management complexity, for example by merging the SGW and PGW of a 4G CN into a single package, or by splitting them into smaller function blocks for reuse and faster response time. Moreover, the effective deployment of VNF instances at the carrier level should be transparent for E2E services.
NFV introduces the following three major differences compared to current practice [29]:

- Separation of software from hardware: it enables the software to be separated from the hardware.
- Flexible network functions deployment: NFV is automatically able to deploy network function software on a set of hardware resources that may run different functions in different data centers at different times.
- Dynamic operation: network operators can scale the NFV performance dynamically and with great flexibility due to the decoupling of the functionality of the network function into the new considered insatiable software component.

The NFV architecture contains three main functional blocks which are: NFV Orchestration and Management (NFV-MANO), NFVI and Services, as shown Figure2.6. These components are defined as follows:

- NFV-MANO: NFV Management and Orchestration (MANO) is composed of the orchestrator, the VNF managers, and the virtualized infrastructure managers. These blocks provide the functionality required for management tasks applied to VNFs, such as provisioning and configuration. NFV-MANO includes orchestration and lifecycle management of physical or virtual resources that support infrastructure virtualization and VNF lifecycle management. It also includes databases used to store the information and data models that

define both deployment and lifecycle properties of functions, services, and resources [7].

- NFVI: the NFV infrastructure covers all the hardware and software resources that comprise the NFV environment. NFVI includes network connectivity between locations, such as between data centers and public or private hybrid clouds. Physical resources generally include computing, storage, and network hardware, providing processing, storage, and connectivity for NFVs through the virtualization layer located directly above the hardware and summarizes the physical resources. The NFV architecture can take advantage of an existing virtualization layer, such as a hypervisor, with standard functionality that simply abstracts the hardware resources and allocates them to the VNFs. When this support is not available, the virtualization layer is often achieved through an operating system that adds software to a non-virtualized server or implements a VNF as an application [7].

- Services : a service is a set of VNFs. It can be deployed in one or more virtual machines. In some situations, VNFs can run in virtual machines installed in operating systems or on hardware directly. They are managed by either native hypervisors or virtual machine monitors. A VNF is typically administered by an EMS (Element Management System) responsible for its creation, configuration, monitoring, performance and security. An EMS provides the essential information required by the Operational Support System (OSS).The OSS is the overall management system that, together with the Business Support System (BSS), helps providers deploy and manage multiple E2E communication services. NFV specifications mainly aim to integrate with existing OSS/BSS solutions [7].

2.3 Network slicing

Network slicing is a recent technology that presents a key element in the new 5G network generation. It represents a research focus at both academic and industry sectors, where several research areas have defined it. In the context of 5G, network slicing is defined by the Next Generation Mobile Network (NGMN) Alliance in [30], as a technology that enables the integration of logical and/or physical network and cloud resources into an open software-oriented multi-tenant programmable network environment. It consists of placing several self-contained logical networks on a common physical infrastructure platform, hence activating a flexible ecosystem of stakeholders that promotes technical and business innovation. 3GPP introduces network slicing as a technology that allows the operator to create customised networks, providing optimised solutions for different requirements in different market scenarios [31]. For ITU-T, network slicing is the sharing and isolation of virtual

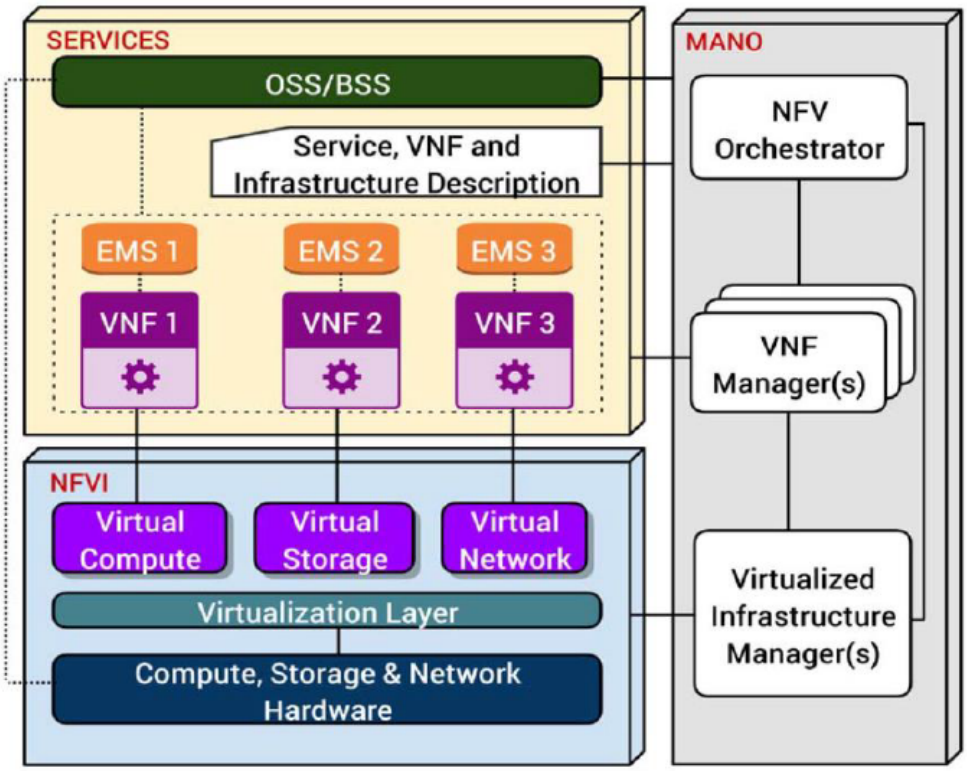


Figure 2.6: NFV architectural framework [7]

resources based on a programmable control and data plane [32].
In general, network slicing provides networking, radio, and virtual resources (i.e., VNF), leading to real service differentiation and customized network operation. Therefore, it enables value for application providers, vertical segments, and third parties without a physical network infrastructure. A network slice consists of consists of a set of VNFs, which can vary significantly depending on the service requirements of that particular slice. The number of resources in each slice depends on the type of service associated with that slice. For instance, a network slice that contains the video download services will receive the appropriate resources and service processing to meet the requested download speed and rate [10].
Network slicing is based on many conditions and principles. The most important and required are as follows [10]:

- Automation: the creation of a network slice in an automatic way without the need of manual intervention or fixed contractual agreements. This is according to its desired requirements, citing: SLA, latency, flow, its duration etc.

- Programmability: allows the control of network and cloud resources of the allocated slice via open APIs to facilitate resource elasticity and service-oriented customization on demand.

- Isolation: represents a crucial property of the network slicing. It allows isolating the tenants from each other despite using the same physical infrastructure to provide their various services with a certain level of security and better performance. Although the implementation of isolation defines the degree of separation of resources in the data plane, it should also be included in the control plane. Isolation can be deployed: (i) in shared resources in the case of separation through virtualization, (ii) by using a different physical resource, and (iii) by sharing a resource based on a respective policy that defines access rights for each tenant.

- Elasticity: is an operation designed to ensure the required SLA under changing conditions. It is related to the resource allocated to a particular network slice according to the number of service users, the mobility of users, the radio, and the network in general. This resource elasticity can be obtained by (i) adjusting the applied policy, (ii) reprogramming the functionality of certain elements of the data and control plane, (iii) relocating the VNFs or varying their number, or (iv) varying the number of resources allocated to each slice, while taking into account the total capacity and the requirement of the other slice requests in terms of resources.

- Customization: ensures the efficiency of the resources allocated to a particular tenant to meet the relative service requirements. The customization of slices can be performed: (i) at the network level, considering the data separation and control plane and the abstract topology, (ii) on the control plane by introducing programmable protocols, operations, and policies, (iii) on the data plane with service-adapted network functions and a data transmission mechanism, and (iv) through value-added services like key data and context awareness.

- Hierarchical abstraction: is a property that has its roots in recursive virtualization. It allows the network slice resources allocated to a particular tenant to be partially or totally traded to another third actor, according to a hierarchical pattern. In this way, the network slice tenant facilitates another network slice service on top of the previous one.

- E2E: it is a property that ensures an E2E service from the service providers to the end-user/customer. It spans different administrative areas, i.e., a slice that combines resources belonging to separate InfProvs. In addition, it unifies heterogeneous technologies and different network layers, for instance, CN, transport, RAN, and cloud. In particular, an E2E network slice consolidates various resources enabling a superimposed service layer. Hence, it provides new opportunities for convergence of services and efficient networking.

In the next three subsections, we will detail network slicing in the CN, RAN,

and transport network. In addition, we will present the architecture of network slicing.

2.3.1 Slicing the Core Network

5G should address the increased demand for throughput, as well as provide greater elasticity, scalability, and flexibility. To this end, the CN architecture defined by previous generations should be more adapted to these new 5G network requirements. To this end, 3GPP in [15] introduces a new CN architecture (namely the Next Generation core or 5G system architecture), where they reshape the EPC entity into more fine grained network functions. This new 5G core, as defined by 3GPP, utilizes service-based architecture, cloud-aligned, that covers all 5G functions and interactions, including security, authentication, session management, and aggregation of traffic from end devices. In addition, the 5G new core emphasizes NFV as an integrated design concept with virtualized software functions that can be deployed using the MEC infrastructure [10].
The new generation core architecture defines some new network functions that did not exist in LTE, while it retains other functions as they were presented in LTE. In particular, session management and access control became separate in the new generation core instead of being combined in the EPC, to better support fixed access and ensure flexibility and scalability [10].
The main network functions defined in the new CN are as follows:

- Core Access and Mobility Management Function (AMF) to manage the access control and mobility amongst others and integrate network slice functionality as part of its basic set of functions.

- User Plane Function (UPF) can be implemented according to the type of service in several locations and configurations.

- Session Management Function (SMF): to manage user sessions according to network policy.

- Policy Control Function (PCF): It corresponds to the PCRF in LTE, where it integrates a policy framework for network slicing.

- NF Repository Function (NRF) this function provides discovery functionality and registration, enabling NFs to discover each other and communicate via open APIs.

- Unified Data Management (UDM), which is similar to LTE's HSS, but with additional subscriber information for mobile and fixed access in the new CN.

The new core architecture is supposed to be deployed within two phases [10]:

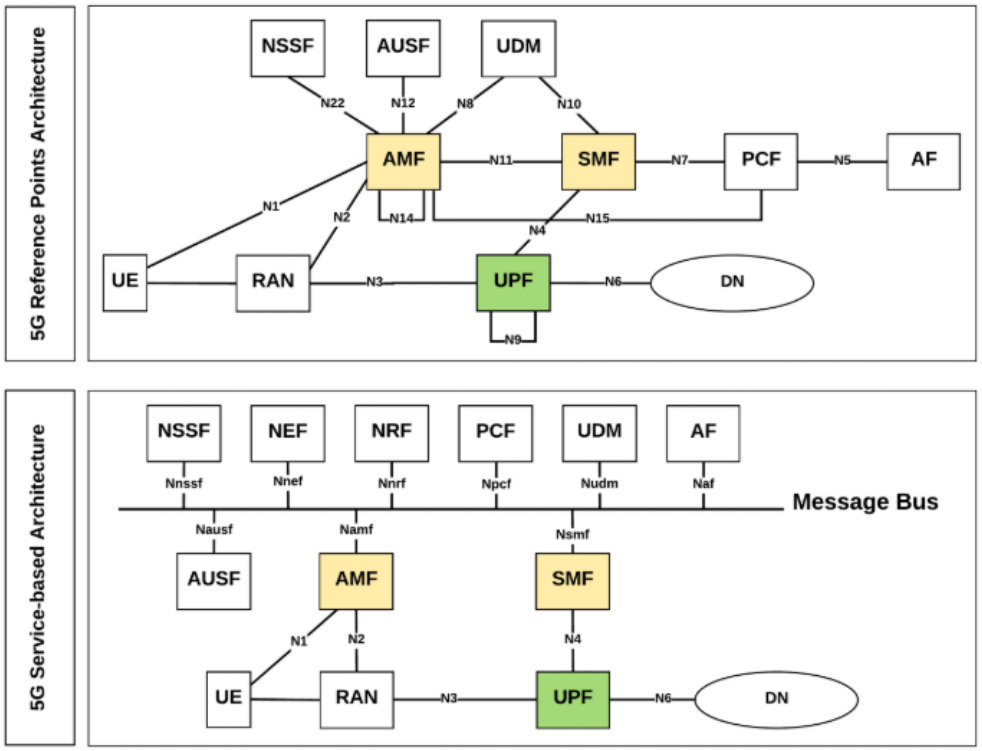


Figure 2.7: new core architecture [8]

- Phase one - 5G reference points architecture: in this phase, the components are connected using a point to point connection, based on reference interfaces, not too much different from the LTE architecture. As shown in Figure 2.7, the 5G CN functions are connected together and with RAN and UEs via reference interfaces. However, this architecture requires a reconfiguration of multiple E2E interfaces from the operators, which may add complexity to add new network elements/instances.

- Phase two - 5G service-based architecture: represents an evolution of the first phase where it differs only at the control plane level, whereas it integrates the same functional elements and the same user plane processing path between the external data networks and the UE. In this architecture, a service model is used to query an NRF to communicate with and discover each other, instead of using the predefined interfaces between elements. Therefore, an NF may register for specific events provided by another NF via an API. This registration is based on the concept of producer/consumer, where 1:N communication becomes possible. The service-based core architecture could be easily deployed in a virtualized and a software environment (e.g., container or VM). The libraries of functions can be retrieved from a VNF catalog and assembled in chains of E2E on-demand services. In addition, the composition of specific functions would allow the 5G CN to be adapted to a particular

network slice.

The fine-granular NFs for network slicing enable certain NFs to be shared between the concerned slices in order to: (i) manage common hardware (if NFs cannot be deployed in a software environment), (ii) reduce signaling load over the air by increasing the number of shared CN control plane NFs, and (iii) reduce the complexity of the management of the network slices, by sharing the UDM and the mobility management procedure. Indeed, it is necessary to identify the common NFs that need to be shared by multiple E2E network slices. This takes advantage of the redesign of CN and RAN functions (i.e., fine granularity) due to the new architecture of core and functional split. The functional split is a mechanism that will be detailed in the RAN slicing sub-section afterward.
It is worth noting that the slice consumer requests a slice creation from the slice provider based on the SLA agreement between them, which includes several conditions of service, including degrees of control and the level of exposure that depends on the slice requirements. The most important control and exposure levels are the following:

- Basic Level Control: This level of control allows only the slice consumer to exercise a passive form of control over the network slice.

- Extended Control and Management: This level of control contains configurations that allow the slice consumer to update the slicing functionality by reducing or increasing them.

- Full Control and Management: In this level of control, the slice consumer has the highest level of network slice privileges and control available. The slice consumer can deploy any form of network functions required by the network slice to provide the desired set of network services. In fact, here, the slice consumer has the right to operate and manage its virtualization platform and other related management support systems.

For this purpose, the slice provider, in order to facilitate the management of their slices requests, create layers of abstraction contained interfaces that map the different levels of control of the network slices exposed to the slice tenants/owners.

2.3.2 Slicing the transport network

5G requires an overhaul of the transport networks in order to satisfy the increasingly bandwidth-intensive requirements of networks. The evolution of the transport network is based on the integration of the backhaul and fronthaul segments on the same transport substrate and the incorporation of deep programming in the transport network.

A. Fixed Access Network Sharing

Fixed Access Network Sharing (FANS) consists of sharing the physical network into multiple virtual network slices and allocating them to different operators, with a high degree of flexibility, based on network virtualization technologies [33].
Two models of a solution have been presented by the introduced fixed access network sharing, as follows:

- Management System, which introduces the network slicing at the management level. The management system configures a virtual private network (VPN), and installs a policy to ensure the appropriate capacity to allocate across the corresponding network equipment.

- Virtual Access Node (VAN), which is based on the function virtualization principles relying on the NFV MANO paradigm. Here, the resources are extracted in multiple VAN instances with virtual to physical port states maintained by a mapper in the management system of the operator. Network slice requests are received via the operator's management system, which communicates with MANO to configure network services and connectivity. Then, using a hypervisor, the VAN allocates network resources to virtual operators taking into account existing cloud computing platforms and routers.

B. Fronthaul/Backhaul and network slicing

The emerging 5G backhaul/fronthaul must be suitable to meet the 5G requirements on integrating various transmission technologies such as optical Ethernet, IP, millimeter-wave. Besides, the 5G RAN must support other emerging requirements that need high-performance and stable connectivity, for instance, adopting massive MIMO or dual connectivity and centralized management. This involves the coordination of the radio and transport layers, tighter synchronization, and unified transport control. In addition, a flexible split of base stations is essential here while taking into account the requirements of the use case and network conditions. In fact, centralization requires higher backhaul/frontier capacity, resulting in increased costs. The split of base station functions has an impact on the role of backhaul and fronthaul, creating an integrated transport network architecture that combines the two, flexibility and reflecting of RAN centralization.
In this context, the authors of [34] [35] introduce an architecture based on the creation of an overlay network. It is an integrated upstream and downstream transport architecture based on a unified control plane and a data plane. It relies on network nodes capable of integrating different upstream and downstream transport technologies via a common data frame. The authors of [36] proposed an architecture based on integrating an optical and wireless backhaul/fronthaul. The data plane here can be programmable according to the SDN paradigm.
Therefore, different service-oriented functional distributions of the RAN on the

same network infrastructure can be adapted. In fact, the functional distribution of the RAN and the degree of centralization depends on the network conditions and the service requirements. Besides, to guarantee the performance and ensure isolation between different logical networks that use a different functional RAN distribution, integrating network slicing can be a key enabling that. To achieve this, network operators use algorithms of slicing and perform a joint optimization of VNF embedding according to the availability of links and the topology of the transport network. It is based on knowledge of the user's context, the network, and the service, to provide the required service.

2.3.3 Slicing the Radio Access Network

The bandwidth or frequency spectrum resources limitation requires the RAN to apply suitable properties to best manage the resource gap. As already mentioned, network slicing should ensure a certain level of service efficiency. The RAN domain mainly needs to guarantee a good customization, elasticity, and an efficient resource sharing and dynamic isolation algorithm. We will detail the three most important criteria that need to be addressed in the RAN regarding network slicing.

A. Slice Resource Management and Isolation

The slice resource management models depend on the required level of isolation, where each slice uses a frequency spectrum or a set of resources dedicated to it. In the RAN, the isolated dedicated resources are the MAC scheduler, control plane, user plane, and the spectrum. Each slice has access to a percentage of dedicated pRBs and its own instances, which are: MAC, PDCP, RRC, and RLC. The dedicated resource model limits the gain of multiplexing and reduces resource elasticity, despite the fact that it ensures delay and capacity constraints. The shared resource model allows slices to share the control plane, spectrum, and MAC scheduler. In fact, the amount of resources (the NpRBs) allocated to a slice cannot be adjusted by the slice owner, even if they are not used. In particular, the pRBs in the shared spectrum are managed by a common scheduler. This common scheduler allows resources to be allocated to slices, according to a specified policy and other commercial criteria. However, this solution does not totally ensure traffic isolation and quality of service, although it exploits statistical scheduling of physical resources granting elasticity.

Several works in the literature have addressed the issue of spectrum sharing among Mobile Virtual Network Operators (MVNO) in the RAN. The authors of [37] propose an approach to the virtualization of an LTE eNB based on a hypervisor, considering the traffic load and radio conditions. The authors of [38] have detailed the management of wireless resources at the network level for RAN sharing. The concept of Network Virtualization Substrate, allowing flexible allocation of shared resources by modifying the MAC scheduler, has been introduced in [39]. Its purpose

is to reflect the traffic needs of MVNOs by taking into account the corresponding SLA. The authors of [40] propose a resource reservation scheme for LTE networks, covering both scheduler and admission control aspects. This scheme can flexibly allocate the shared resources to operators according to their traffic priorities and actual traffic loads. The paper [41] introduces an application-oriented framework for RAN sharing in mobile networks, according to the applications' QoS requirement.
The majority of the systems proposed in the literature are based on applying optimization models with multi-objective functions. These models aim at satisfying a specific range of heterogeneous slice requirements such as throughput and latency. However, the most of proposed solutions and optimization models are NP-hard and complex. To this end, the authors of [16] introduce a two-level scheduler to share the pRBs between the network slices in order to relax the constraints of the already proposed NP-hard solutions. The first level allows the allocation of virtual RBs (vRBs) to the UEs belonging to a slice; this level is called Slice Resource Manager (SRM). The second level is an inter-slice scheduling process that translates the vRB allocation into PRB; this level is called Resource Manager, considering the resource capacity limitation.

B. RAN Programmability

RAN represents both the most complex and the most expensive part of the mobile network infrastructure. The technologies and strategy adopted to increase system capacity and improve spectrum efficiency require a high level of coordination between base stations. Therefore, integrating SDN in the RAN is seen as a solution allowing a significant evolution towards the future. It enables a wide range of use cases and new services. This concept is called Software-Defined RAN (SD-RAN) or RAN programmability. SD-RAN presents a challenge, given several strict constraints that must be addressed. Among these challenges, there is the time constraint associated with some key RAN control operations. In particular, RAN programmability allows APIs to be easily opened to third parties. In addition, it enables to abstract the underlying RAN resources through a service orchestrator entity. This service orchestrator entity allows to dynamically control the resources dedicated to a network slice.
Softwarization of the RAN has received substantial attention from the research community in recent years in both industry and academia. For instance, the authors of [42] present the idea of the abstraction of a big base station. This solution enables the management of the dense network deployments by separating the data plane and the control plane, which is centralized. The authors of [43] have developed FlexRAN, which is a flexible and programmable SD-RAN platform that separates the RAN control and data planes through a new custom-tailored southbound API. The authors implement FlexRAN as an extension to a modified version of the Ope-

nAirInterface LTE platform [44]. Precisely, FlexRAN represents the first SD-RAN platform, where its design facilitates real-time RAN control due to its flexible architecture. It is transparent to the end devices, which facilitates evolution and deployment. It offers two levels of programmability; the first level is at the controller, which allows updating the implementation of any control function on the fly. The second is a form of RAN control/management applications that can be built on the FlexRAN controller. In this context, the authors of [16] present a new architecture applying FlexRAN in the RAN, based on the two-level MAC architecture already defined by [12]. The authors implemented Open Air Interface (OAI) in a two-level scheduler and demonstrated via FlexRAN the life cycle management of the RAN slice. The authors of [45] design an approach allowing to drive resource disaggregation in centralized software in the RAN. It enables individual allocation of processing functions to different servers according to the type and volume of their processing requirements.
In similar context, several industry consortia have introduced an extension to the RAN called xRAN to design the future programmable wireless dense network infrastructure. The goal is to decouple service from hardware by designing a programmable and generic substrate to create a flexible multi-service network. Recently, Central Office Re-Architected As a Datacenter (CORD) and xRAN have created an open carrier-class reference implementation of xRAN in the context of M-CORD, by combining their efforts. Also, PRAN [46] and OpenRadio [47] address data plane programmability and the deployment of new wireless protocols on the fly, in addition to the control plane solutions already proposed.

C. Flexible RAN Virtualization and Functional Split

RAN virtualization enables certain RAN functions to run on remote software platforms, based on the concept of BS softwarization. This paradigm has undergone significant development with the emergence of the Cloud-RAN (C-RAN) concept [48] [49]. In this context, the RAN functions are split between the remote radio heads (RRH) that provide the antenna equipment and radio access and the baseband unit (BBU), hosted in the cloud. C-RAN deployments are based on the concept of flexible, functional distribution. It considers a range of C-RAN deployment options based on the time-critical nature of certain RAN functions and the front-end capacity while considering environmental conditions and user' load.

In the RAN, the gNB architecture is composed of two main units, namely the: Central Unit (CU) and Distributed Unit (DU). CU is a logical node that includes the gNB functions like radio access network sharing and mobility control. CU controls the operation of DUs over the fronthaul interfaces. On the other hand, DU is a logical node, where its operation is controlled by CU. DU includes a subset of the gNB functions, depending on the functional split option. Fs-C and Fs-U provide control plane and user plane connectivity over Fs interface. Connectivity between

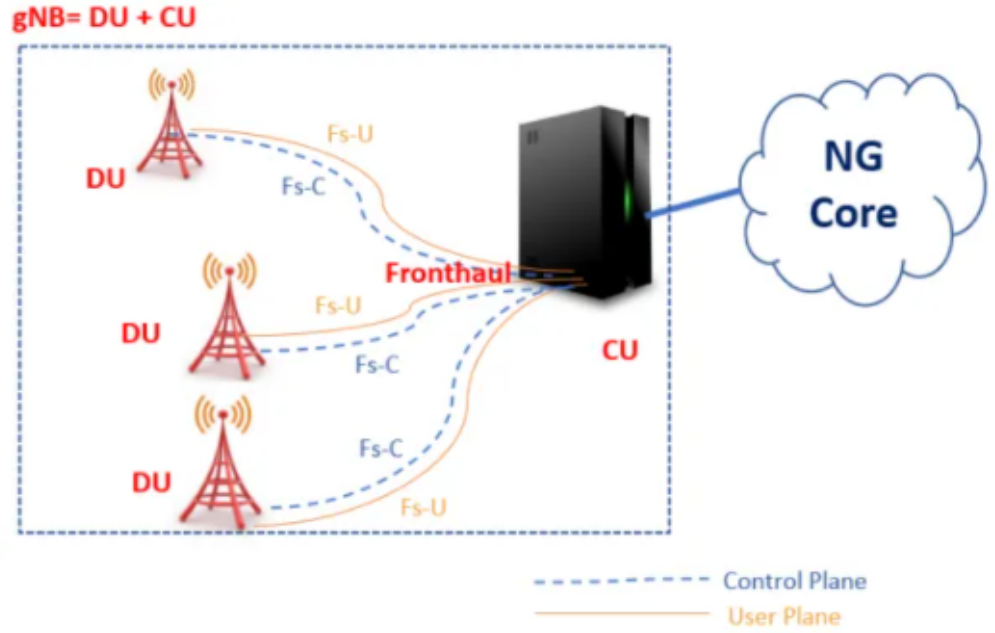


Figure 2.8: gNB Logical architecture [9]

CU and DU is provided via the Fs interface. Figure 2.8 shows the gNB logical architecture, and its CU/DU split.
Figure 2.9 presents an overview of the different CU/DU functional splits options, that are detailed as follows:

- Option 1 (RRC/PDCP split): In this split option, RRC is in the CU while RLC, PDCP, MAC, PHY layer, and RF are kept in the DU. Hence the entire user plane is in the DU.
- Option 2: (PDCP/RLC split): This option can use any type of the first-line network, it runs PDCP functions at the BBU, and it is not time-critical. The

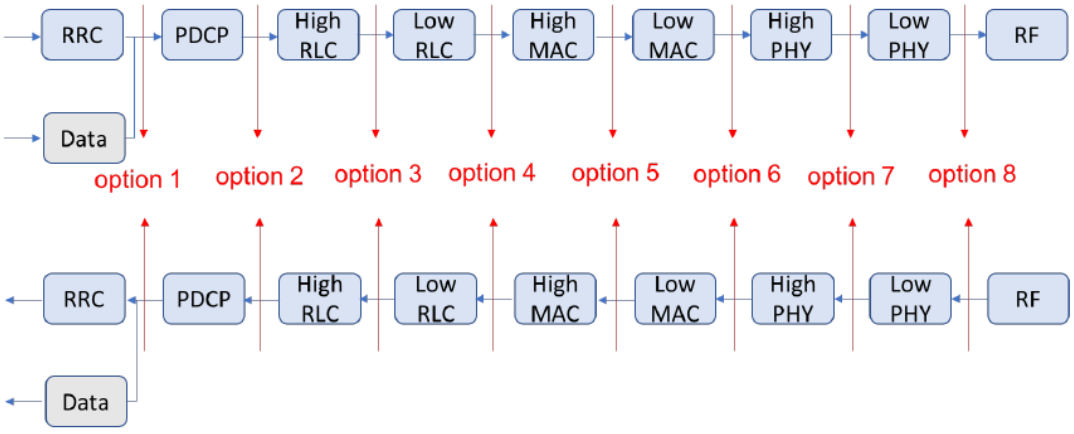


Figure 2.9: Functional split overview [10]

main advantage of this option is the possibility to have an aggregation of different RRHs technologies (e.g., 5G, LTE, Wifi).

- Option 3: (High RLC/Low RLC split, Intra RLC split): In this option, several MAC entities are associated with a common RLC entity since the RLC layer, and other layers above it are virtualized at the BBU. Due to the fact that real-time scheduling is done locally in the RRH, this option reduces the latency constraints of the fronthaul.

- Option 4 (RLC-MAC option): In this split option, RRC, PDCP, and RLC are in the CU. MAC, PHY layers, and RF are in the DU. The MAC and the protocols that precede it are virtualized and run real-time scheduling on a BBU aggregated across multiple RRHs. Although this option allows coordinated planning and dynamic point selection, it requires a low-latency fronthaul. In fact, some MAC procedures need to generate a configuration at the TTI level and are time-critical.

- Option 5 (Intra MAC split): this option assumes the following distribution: (i) Higher part of the MAC layer (High-MAC), RLC and PDCP are in the CU and (ii) PHY layer, RF and lower part of the MAC layer (Low-MAC) are in the DU. The services and functions provided by the MAC layer will be located in the CU, since the MAC layer is splits into 2 entities (e.g. High-MAC and Low-MAC).

- Option 6 (MAC-PHY split): This option offers the highest level of centralization. It can only be achieved through an ideal fronthaul using a low-latency optical fiber and a high data rate.

- Option 7 (Intra PHY split): This option requires a compression technique to reduce transport bandwidth requirements between the CU and DU.

- Option 8 (PHY-RF split): This option separates the PHY layer and the RF layer enabling the centralization of processes at all levels of the protocol layer. It results in very tight coordination of the RAN. Thus, it provides efficient support for functions such as MIMO, mobility, and load balancing.

RAN Spilt architecture, as described above, provides significant advantages, including: (i) real-time performance optimization, load management, and enables NFV/SDN; (ii) adaptation to various use cases; (iii) scalable, cost-effective solutions.
In this context, the IEEE NGFI (Next Generation Fronthaul Interface) introduced in [50] some functional splits considering the interface latency and bandwidth requirements.
It is worth noting that the flexible functional split can highly improve the network slicing performance. This can be achieved by having an optimal split, which

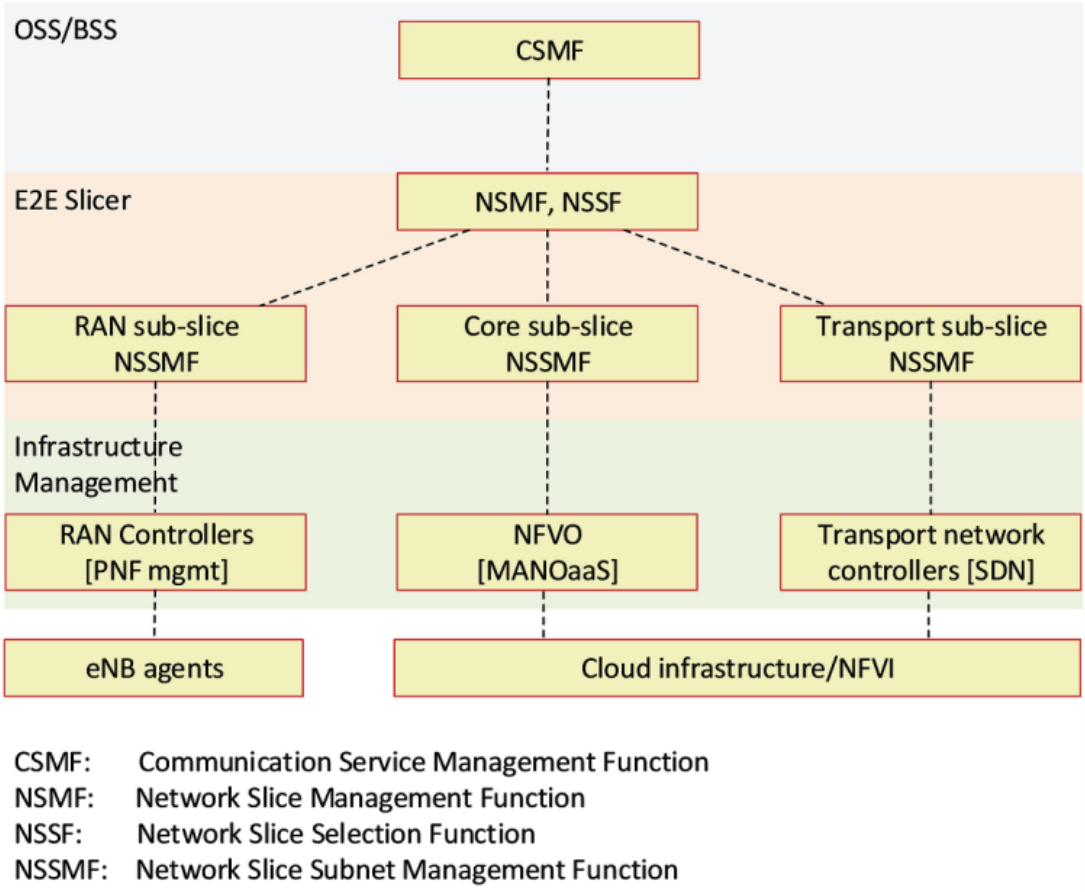


Figure 2.10: E2E network slicing architecture: high-level view [11]

depends largely on the desired service features. For an eMBB slice, a higher centralization can enhance the throughput by aggregating RRHs enabling, while an uRLLC slice may need most RAN functions to run on DU in order to fulfill latency requirements.
In the context of network slicing and functional split, the authors of [51] showed that certain RAN functions can also be shared among different slices. For instance, we can mention the low RLC (real-time function), MAC scheduling (inter-slice scheduler), and PHY layer while dedicating RRC (configured and tailored user plane protocol stack), PDCP, RLC (non-real-time functions) to each slice separately.

2.3.4 Network slicing architecture: orchestration and management

At a higher level, an E2E network slice should always be composed of three subslices: core, RAN, and transport. According to the 3GPP specifications of [52], the authors of [11] present the architecture of the high-level network slicing as shown in Figure 2.10.

In this architecture, the Communication Service Management Function (CSMF) component gets a demand for a communication service by a vertical and translates it into specific slice requirements. Therefore, it requests the instantiation of an E2E network slice by interacting with the E2E slicer through the northbound interface (NBI).
The Network Slice Management Function (NSMF) of the E2E slicer manages the lifecycle of every 5G service slice. Each network slice type request has specific requirements (for instance: number of UEs, slice life duration, etc.), which are provided as a slice template. To this end, an appropriate Network Slice Subnet Management Function (NSSMF) corresponding to each sub-slice will be selected according to the template. Afterward, the NSMF creates a customized slice instance and assigns it to the appropriate NSSMF, the instantiation and management of each sub-slice. Note that the slice template has an important functionality for handling the sub-slice of the CN. Indeed, each slice template contains fields that indicate the NSSMF's type that should be used to orchestrate the underlying sub-slice instance. This functionality allows the NSMF to identify which NFV Orchestrator (NFVO) and the respective MANO stack to launch or reuse one slice instantiation type. It will then request creating the appropriate CN sub-slice instance through the NBI of NFVO, by properly customizing the VNF instances included and the relevant resources depending on the service-specific characteristics.
The RAN sub-slice NSSMF contains a RAN resource allocator component responsible for translating the slice needs into a radio resource allocation and performing a high-level RAN resource allocation. To this end, the RAN state should always be updated in terms of different varying conditions, including the bitrate requirements per UE/slice, the connected UEs per eNB, the quality of their radio connection, and the slice instances to which an eNB is participating. In this case, the NSSMF uses this information to generate an adequate distribution of RAN resources per cell to meet the coexisting slice requirements. The NSSMF uses the appropriate NBI from the RAN controller to ensure changes in radio conditions or deploy new slices. In this context, (i) the RAN controller is considered as a RAN-specific Virtual Infrastructure Manager (VIM), (ii) an agent located at an eNB is considered as a hypervisor, where it provides a virtualized view of the RAN resources, and the necessary primitives to perform resource management tasks on physical resources dedicated to a slice on the same radio hardware.
The transport-level NSSMF interacts with network elements to handle the isolation and the provisioning of the links (physical or virtual) connecting network functions of the RAN and CN and external networks.

2.4 5G network slicing challenges and open issues

A. Intelligent service function chaining

The network slice is composed of a set of service function chains. Service function chaining is a mechanism that links different service functions to each other to form a service through virtual machines running either on a single or multiple nodes. These nodes have specified bandwidth constraints on communication with other nodes and limited computing resources [53] [54] [55] [56].
Therefore, the performance of service function chaining significantly depends on the choice of the node and its corresponding communication links. To this end, an optimal choice of the node and routing protocols is required here to enhance the performance of this service chaining citing, computational latency, energy, BW limitation, etc.
In the literature, there are some works based on optimization algorithms and ML techniques used to address this issue (i.e., service chaining). On one hand, the algorithms based on mathematical optimization have shown their effectiveness in this case. However, most of them are very complex to solve and slow, which requires heuristics, which makes the results less accurate [54] [57]. On the other hand, the use of ML algorithms to solve this problem is a recent trend that shows its effectiveness but still presents some challenges, which must be addressed. For instance, the authors of [58] present a study that enables intelligent service function chaining based on a deep learning scheme and design a high-performance routing strategy in SDN and NFV-enabled networks. The authors of this work only considered latency constraints without considering the energy which should be considered as a joint minimization with the latency More recent papers are published in this context, citing [57], where the authors propose an optimized solution of resource allocation of service function chain in NFV-SDN using a Reinforcement Learning algorithm. The authors of [59] propose a geo-distributed VNF chain over time by modeling the traffic using a recurrent neural network and making chain placement decisions using deep reinforcement learning. The authors of [60] propose a service function chaining scheme based on reinforcement learning.

B. Mobility-aware slicing

Mobility aspects such as interference management and seamless handover pose significant challenges to network slicing. Mainly, these challenges are due to: (i) the increasing number of mobile users of a wide variety of smart applications and to (ii) the handovers to the different access networks, especially for real-time services. For this purpose, a slicing system addressing mobility management issues is essential to deal with mobility challenges, especially for critical services such as automated driving. In this context, the authors of [61] propose a solution to enable mobility-aware network slicing based on the Lagrangian dual decomposition. The authors of

[62] propose to track user mobility and improve the network slicing system utility using the long short-term memory (LSTM) algorithm. The authors of [63] introduce optimal mobility management enhancements by re-evaluating the optimality of the mobility anchor during each handover in order to ensure low latency in 5G network slicing. The authors of [64] propose an architecture using the slicing network paradigm to enable user mobility in different radio access technologies.

C. Dynamic spectrum slicing

Dynamic spectrum slicing represents a crucial and very important challenge in network slicing due to the limited spectrum bandwidth and to the growing demands of users. It is necessary to efficiently utilize the available spectrum via slicing while considering the significant variations of users' demands. Therefore, when the spectrum allocation is fixed without following the users' demands, the spectrum bandwidth maybe under or over-utilized. To this end, it is necessary to define a dynamic spectrum allocation algorithm based on the varying resource demand of users according to their application requirements over time. For instance, the authors of [65] present a scheme of dynamic spectrum policy considering the network traffic variations, where they use Markov process to model the user traffic distribution. The authors of [66] introduce a novel solution to address effective spectrum slicing to address the dynamic data requirements at every 5G base station. They propose the application of overlapping coalitional game models to spectrum slicing among operators. In the same context, the authors of [67] introduce a service-oriented spectrum-aware RAN slicing trading model in order to ensure a dynamic on-demand RAN-slicing under the spectrum sharing scenario based on mixed-integer nonlinear programming.

D. Resource sharing

Sharing resources between slices presents a crucial issue. It can be static or dynamic. The algorithm of resource sharing should follow the dynamic requirements of each slice based on several constraints citing, the applications running in the slice, the slice required throughput, the slice required latency, the number of users hosted by the slice, and others. Therefore, a dynamic resource sharing algorithm is more efficient and recommended than a static one to fulfill the dynamic slice requirements. In addition, resource sharing introduces other challenges that must be considered, such as the slice isolation, despite all the benefits it brings to the InfProv. It is worth noting that the resource sharing among the RAN slices is more critical and challenging compared to core slices.
The authors of [68] present a general study on the resource sharing existing mechanisms in network slicing. In this context, several other works in the literature proposed algorithms to deal with resource sharing challenges [69] [70] [71].

E. Isolation among network slices

Slice isolation presents a critical challenge in network slicing. In fact, it allows enforcing the core concept of network slicing concerning the simultaneous coexistence of several slices sharing the same infrastructure. Different services in 5G networks have unique requirements. Therefore, dedicated virtual network resources are required to guarantee the service quality at each slice. Efficient slice isolation should ensure that any security attack or failure on one slice will not affect the operation of the other slices. In addition, it should ensure the privacy of each slice.
In the literature, some works have addressed this network slicing isolation issue. For instance, the authors of [72] present the problem of isolation-aware RAN slice mapping, considering a 3-layer RAN architecture and advanced functional splits, using a dual-objective heuristic algorithm. The authors of [73] propose to use an appropriate user admission control mechanism to ensure isolation in dynamic network slicing. In [74], the authors demonstrate that Flex Ethernet technology is able to guarantee physical isolation for virtual networks and avoid interference between different slices.

F. Algorithmic aspects of resource allocation

The design of a good slice resource allocation algorithm is a difficult challenge that needs to be addressed. It is worth noting that this problem is similar to the virtual network embedding problem. These algorithms can be based on mathematical methods of operations research, such as integer linear programming while considering the size of the slice. In addition, the performed algorithm should be able to reconfigure and migrate slice resources due to the dynamic characteristic of the 5G network. In this context, the authors of [75] present the algorithmic challenges that arise in efficient network slicing, including the resource allocation aspect. The authors confirm that these algorithms require new techniques from computer science, operations research, and networking.

Chapter 3

Dynamic slicing of RAN resources for heterogeneous coexisting 5G services

Network slicing is one of the key components allowing to support the envisioned 5G services, which are organized in three different classes: eMBB, mMTC, and uRLLC (refer to Fig 3.1). Although it is straightforward to slice and isolate computing and network resources for CN elements, isolating and slicing RAN resources is still challenging. In this chapter, we leverage on the two-level MAC scheduling architecture [12] to provide a resource sharing algorithm that computes and dynamically adjusts the necessary radio resources to be used by each deployed network slice, covering eMBB and uRLLC slices. We also present the simulation results of the proposed algorithms for resource sharing between slices, which clearly indicate the proposed solution's ability to slice the RAN resources and satisfy the heterogeneous requirements of both types of network slices.

3.1 Motivations and state of the art

Network slicing aims at sharing the same physical infrastructure (Mobile Network infrastructure, RAN, and CN) by creating virtual instances of the network tailored to application needs. Network slicing requires sophisticated mechanisms to share and isolate the RAN resources across slices. In [12] and [16], the concept of a *two-level scheduler* is introduced, which aims to share physical radio resources (i.e., *pRBs*) among slices by abstracting pRBs and using two scheduler levels as shown Figure 3.2. The first level is slice-specific, allowing each slice to use its own internal scheduler, and schedules each UE with vRBs. On the other hand, the second level considers the slice-specific (virtual) resource assignment and maps it to actual pRBs. As the number of pRBs (N_{pRB}) is limited, the second-level scheduler con-

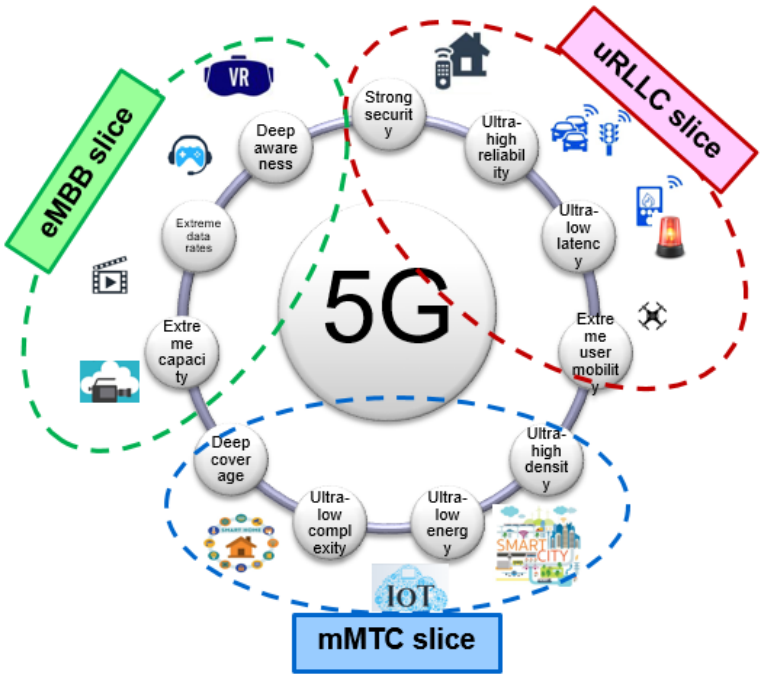


Figure 3.1: Slices defined

trols the number of N_{pRB} assigned to each slice according to the recommendation of a *SO*. The latter indicates the maximum N_{pRB} to dedicate to each slice after executing an intra-slice physical resource sharing algorithm.
However, in the above works, it is not detailed how these values (N_{pRB}) are derived for each slice, knowing that each slice type has its own characteristics and requirements. For instance, eMBB requests high bandwidth while uRLLC aims to minimize latency and maximize reliability. Rather, the ratio of radio resources to allocate to each slice is considered static and is decided in a manner agnostic to the actual application requirements of each slice in terms of latency and/or throughput. In this thesis, our first contribution is presented in this chapter and fills this gap by completing the work of [12] and [16] with a dynamic RAN resource slicing mechanism to derive the value of N_{pRB} to dedicate to each running slice, according to its specific requirements and the varying conditions of the radio environment. The proposed mechanism runs at the SO level and relies on monitoring information obtained from the RAN.

In this context, several works in the literature address the allocation of resources to network slices. In [76], the authors discussed the dynamic allocation of RAN resources to different tenants (e.g., virtual mobile network operators and service providers). They proposed a weighted proportionally fair allocation mechanism to ensure the desirable fairness and protection among the network slices of the different tenants and their associated users. The authors of [77] designed optimization algorithms for common scheduling between eMBB and uRLLC slice traffic, considering the dual objectives of maximizing utility for eMBB traffic while satisfying instanta-

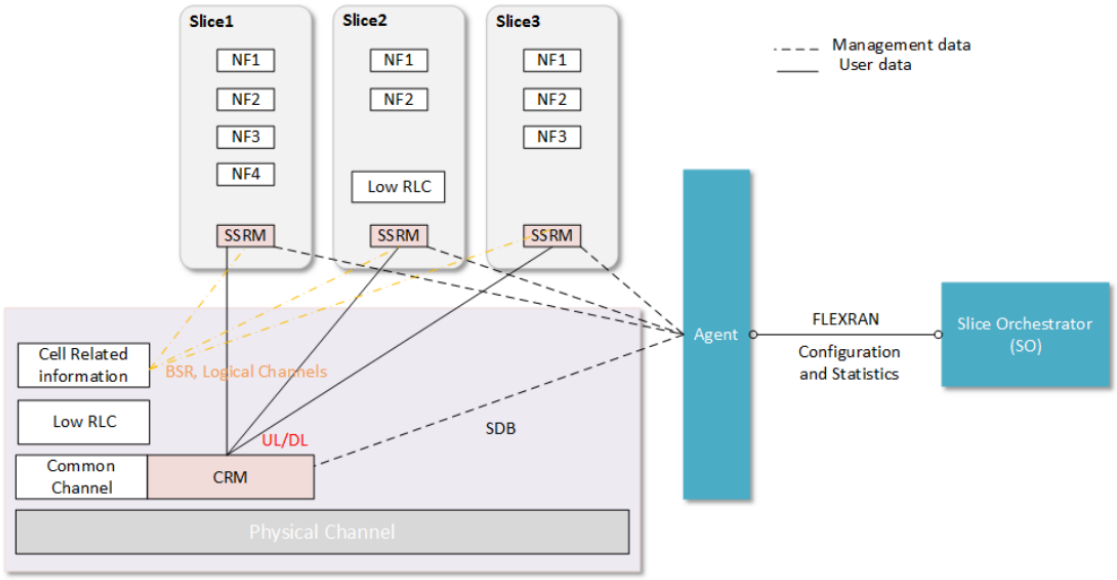


Figure 3.2: 2 leval MAC scheduler system model [12]

neous uRLLC requests. This is achieved by dynamically multiplexing the uRLLC traffic through puncturing/superposition of the eMBB Traffic. The results showed that this joint problem has structural properties that enable clean decomposition and corresponding algorithms with theoretical guarantees. In [78], the authors analyzed dynamic resource sharing in network slicing when tenants (such as mobile operators and/or services) support inelastic users with minimum rate requirements. They proposed a network slicing framework combining (i) admission control, (ii) resource allocation, and (iii) user dropping, which they study using tools from game theory. The authors of [79] presented algorithms that study the problem of resource allocation in the context of a slicing-ready 5G network. These algorithms are composed by: i) traffic analysis and prediction per network slice using the Holt-Winters forecasting procedure to analyze and predict future traffic requests associated with a particular network slice, ii) admission control decisions for network slice requests using a heuristic algorithm, and iii) adaptive correction of the forecasting solution based on the measured deviations, using a proposed network slice scheduler. The authors of [80] focused on the computational outages that can occur between RAN functions, aiming to improve the performance of scheduling and MCS selection functions. The problem, which was shown to be NP-hard, was formulated as a joint optimization one, and some algorithms to solve it were proposed. Finally, [81] adopted revenue management models, which have been introduced in other contexts (airlines, hotels, etc.), in order to propose a resource allocation model. The authors proposed the concept of slice overbooking to maximize mobile operators' revenues by introducing a hierarchical control plane to manage the orchestration of slices.
To summarize, despite the fact that all the methods already proposed have shown relatively good results in the challenge of dynamic resource allocation, all these

methods proposed algorithms for dynamic resource sharing individually or in combination, which are based on several constraints on users, operators, etc. requiring information from them, and which can not always be feasible, optimal and/or accurate. However, the contribution introduced in this chapter proposes a simpler algorithm based only on the estimation of the quality of the channel, which allows predicting the number of resources to allocate to each slice, which adds more precision to the system.

3.2 Architecture and assumptions

In this contribution, we envision the same network architecture model adopted in [12] and [16]. We assume a 5G network which includes a SO and a set of eNBs deployed covering an area. The role of the SO is to deploy and manage the life cycle of network slices in the mobile network (RAN and CN). We assume that a SO is responsible for a region covered by a certain number of eNBs. The SO communicates with the eNBs using a southbound protocol, such as FlexRAN [43], that allows to interact and manage remotely the eNBs. The eNB management process consists in getting status information on the RAN and appropriately configuring eNBs, e.g., by setting the N_{pRB} to dedicate to each slice. We assume that a set of UEs are served by/associated with a network slice, spanning a set of eNBs (i.e., different physical locations). The SO receives from a tenant (owner) a request to instantiate a slice in the form of a slice template, which indicates the slice type (e.g., eMBB, uRLLC, mMTC), its duration, the list of involved UEs, the (application) data rate (denoted by λ) of the service used in this slice, and application requirements such as the maximum tolerated latency. According to this information, the SO derives the appropriate number of $pRBs$ that fits the needs of the slice, which will be communicated later to the involved eNBs via the southbound protocol.
In this contribution, we consider that a network slice is either eMBB or uRLLC; hence, we propose two corresponding mechanisms to estimate the N_{pRB} needed by each slice. Note that although the 5G system considers three types of slices, where, we considered only two of them; resource allocation for eMBB and mMTC may follow the same mechanisms and algorithms. The main difference lies between eMBB and uRLLC, as the first one seeks high data rate, while the second requires low latency. The proposed algorithm first derives an initial estimation of the N_{pRB} necessary using the information obtained from the slice template. Then, a dynamic algorithm is used to tune N_{pRB} periodically according to the feedback obtained from the eNBs via the southbound protocol. In this section, we will detail the first step for each network slice type considered (eMBB and uRLLC).

3.2.1 eMBB slice

An eMBB slice requires a high data rate, representing the main objective when estimating N_{pRB}. In the first step, we start by estimating the maximum number of required $pRBs$ for each eNB i ($N_{pRBmax}(i)$), using the information provided by the slice owner, i.e., the data rate per user required by the application running on top of the slice ($d_{App/user}$), and the number of users ($N_{users}(i)$) of the network slice connected to eNB i, which can be retrieved from the eNB via the southbound control protocol. The main constraint to satisfy is that the number of pRBs $N_{pRBmax}(i)$ to dedicate periodically for an eMBB slice at each eNB should be (greater than or) equal to the aggregate data rate needed by the slice application for all users connected to it; this is captured in (3.1).

$$N_{pRBmax}(i) * d_{pRB} = N_{users}(i) * d_{App/user}. \tag{3.1}$$

Indeed, the equation indicates that the $N_{pRBmax}(i)$ allowed to a slice on a given eNB i should cover the needed slice's applications (i.e., the number of active users $N_{users}(i)$ multiplied by the data rate required by the application). Here we consider that $d_{App/user}$ is the same for all users. We further assume that d_{pRB} is the maximum data rate provided by one pRB and that it is the same for all users. In this first step, we consider that this rate is the maximum achievable by the radio system for ideal channel conditions, i.e., the maximum possible CQI value of 15, and the corresponding MCS and transport block size as specified in the standard [82].
Once each $N_{pRBmax}(i)$ is computed using (3.1), it is communicated via the southbound protocol to the corresponding eNBs.

3.2.2 uRLLC slice

Knowing that a uRLLC slice includes all services requiring ultra-low latency, the aim when deriving N_{pRBmax} is to keep latency below a maximum threshold (Lat_{max}) indicated in the slice template provided by the slice owner. To do that, we need to derive a model that estimates the latency experienced by uRLLC packets at the eNB queue.
Since each slice has its own DL queue at the eNB [12], all packets belonging to the slice share the same queue. Therefore, to estimate the latency of the packets, we propose to model the slice queue at the eNB as an M/M/1/K one. The traffic arrival rate follows a Poisson distribution with intensity λ, the service rate μ is exponential, and the queue has a size of K. Here, the value of λ corresponds to the traffic rate of the application running on top of the slice, while the service rate μ depends on the scheduling process at the MAC layer. To derive λ and μ, we use the following formulas:

$$\mu = \frac{N_{pRB} * d_{pRB}}{avg\_packet\_size} \tag{3.2}$$

$$\lambda = \frac{N_{users} * d_{App/user}}{avg\_packet\_size}, \tag{3.3}$$

with avg_packet_size denoting the average packet size of the uRLLC application, and $d_{App/_user}$ having the same value for all slice users. To estimate the latency of uRLLC packets, we apply Little's law. The latter assumes that whatever the distribution of the arrival rate, the average time a user spends in a queue depends on the number of active users N_{users} and the traffic intensity (i.e., λ). As the number of users corresponds in our case to the number of packets (N_{packet}) of the uRLLC service waiting in the queue, Little's law is used as follows to derive the time a packet spends in the queue:

$$T_w = \frac{N_{packet}}{\lambda} \tag{3.4}$$

As we assumed that the uRLLC queue is modeled as M/M/1/K, N_{packet} can be derived as follows:

$$N_{packet} = \frac{1-\rho}{1-\rho^{K+1}} \sum_{k=0}^{K} k\rho^k \tag{3.5}$$

where $\rho = \frac{\lambda}{\mu}$. Since μ corresponds to the service rate of the uRLLC queue, and depends on the number of resources dedicated to the uRLLC slice, it can be derived using (3.2). By assuming that Lat_{max} is the maximum tolerated latency by a uRLLC slice, T_w should be less than or equal to this value:

$$T_w \leq Lat_{max}. \tag{3.6}$$

We substitute T_w by its value given by (3.4), obtaining the following expression:

$$\frac{N_{packet}}{\lambda} = \frac{\frac{1-\frac{\lambda}{\mu}}{1-(\frac{\lambda}{\mu})^{K+1}} \sum_{k=0}^{K} k(\frac{\lambda}{\mu})^k}{\lambda} \leq Lat_{max} \tag{3.7}$$

Therefore, we need to find a value of μ, noted μ_{opt}, that ensures at least a latency equal to Lat_{max} for uRLLC. According to (3.2), we can extract the number of pRBs (noted N_{pRBopt}) to dedicate to a uRLLC slice as follows:

$$N_{pRBopt} = \frac{\mu_{opt} * avg\_packet\_size}{d_{pRB}} \tag{3.8}$$

At this step, we go by the assumption that the value of d_{pRB} is the same for all UEs, as in the case of eMBB, and that avg_packet_size is constant, and aim to solve (3.7) for μ. We denote the solution to (3.7) as μ_{opt}.
Deriving μ_{opt} analytically is not straightforward. Therefore, we numerically estimate it using the following simple algorithm.[1]

[1]Adaptations of standard numerical techniques such as the Newton-Raphson and the bisection algorithms are also applicable.

Algorithm 1: Calculation of μ_{opt} that allows to respect the latency requirement of a uRLLC slice.

Result: μ_{opt}
initialization: $Mu = [\mu_1, \mu_2, ..\mu_L]$, $M_{opt} = []$
for $l \leftarrow$ *1:L* **do**
  $\rho(l) = \frac{\lambda}{Mu(l)}$
  $N_{packet}(l) = \frac{1-\rho(l)}{1-\rho(l)^{K+1}} \sum_{k=0}^{K} k\rho(l)^k$
  $T_w(l) = \frac{N_{packet}(l)}{\lambda}$
  if $lat_{max} - T_w(l) \geq \epsilon$ **then**
    M_{opt}.append($Mu(l)$)
  else
    reject $Mu(l)$
  end if
  =0
end
$\mu_{opt} = \min M_{opt}$

The steps of this procedure are as follows: First, we generate L candidate values for μ and keep them in a vector Mu. The number of values to generate is limited: For example, since $dpRB$ is assumed for now fixed, we can generate one μ value for each possible number of pRBs, which is defined by the available bandwidth for the given radio technology (e.g., for a bandwidth of 5Mhz, a maximum of 25 pRBs can be used) using (3.2). Then, we calculate N_{packet} corresponding to each value of μ and the resulting T_w value, which we compare with Lat_{max} to check if condition (3.6) is respected.
Note that we use a latency margin ϵ when we compare T_w with Lat_{max} to accept or reject a μ value. By appropriately controlling ϵ, we can ensure that T_w is adequately lower than the latency threshold Lat_{max}, but also close enough to it in order not to waste a lot of resources while respecting condition (3.6).
Out of all the μ values that lead to an acceptable latency (in case there are multiple), we select as the optimal the one which minimizes the difference between T_w and Lat_{max}, i.e., the smallest value of M_{opt}. These steps are illustrated in Algorithm 1. Once μ_{opt} is obtained, we use (3.8) to derive the corresponding N_{pRB} to be assigned to a uRLLC slice. As for the case of eMBB, the proposed method needs to be run for each eNB where UEs of the slice are connected to.

3.3 A channel quality-driven algorithm for dynamic N_{pRB} estimation

The initial N_{pRB} calculated per slice in the previous step is based on the assumption that d_{pRB} is fixed for all users. However, users experience different channel conditions, and hence different data rates.
Therefore, we propose to correct the estimation of the d_{pRB} by using per-UE channel quality reports obtained from eNBs. These reports include the CQI and MCS values of each UE belonging to a cell. Note that these values are transmitted to eNBs by the UEs in order to be used in the scheduling process. We organize these CQI values in a matrix $v(j,k)$, where j is the id of the slice and k is the id of the UE. Based on the CQI, we can estimate d_{pRB} per UE and per cell (eNB). Indeed, d_{pRB} can be obtained by using the same tables used by the eNB to translate a CQI to a data rate [82]. Matrix v is then transformed to a matrix of data rates noted $d_{pRB}(j,k)$, where j and k have the same meaning as for matrix v.
Algorithm 2 presents the different steps of the dynamic slice resource allocation procedure. Note that $Slice(j)$ gives the type of the deployed slice, $N_{pRBopt}(i,j)$ is a matrix that gives for each cell i the necessary number of $pRBs$ for slice j, $N_{users}(i,j)$ a vector indicating the number of users of a slice j in cell i, and $d_{App\_user}(j)$ the data rate required by an application (per user) running on top of a slice j. This algorithm allows to estimate the N_{pRB} allocated to each slice and for each network cell more accurately: For an eMBB slice, it sums the necessary resources per UE considering each user's individual radio capacity reflected in $d_{pRB}(j,k)$. For uRLLC, it applies (3.8), using the optimal service rate as computed by Algorithm 1 to attain latency requirements, and the mean achievable d_{pRB} across all slice users per eNB considering each user's channel quality, instead of a fixed optimistic value for all.
Note that this algorithm is run periodically by the SO. It relies on the eNBs' reports also obtained periodically. The periodicity of running these algorithms is independent from the scheduling period TTI used at the MAC layer of the eNBs.

3.4 Performance Evaluation

3.4.1 Scenarios and parameters

To evaluate the performance of the proposed solution, we extended the Matlab implementation of the two-level scheduler used in [12]. We mainly modified the SO part to include our algorithms. In this simulation, we considered two types of slices, i.e., eMBB and uRLLC. Each slice is defined by the required application data rate, the number of users, the maximum latency for uRLLC, etc.
We simulated different scenarios, where we varied the number of users of the uRLLC slice ($N_{uRLLCusers}$) while keeping it fixed for the eMBB slice ($N_{eMBBusers}$) to 5 users, and for different channel qualities: (i) medium quality where the CQI varies from

Algorithm 2: Calculation of N_{pRB} for eMBB and uRLLC slices for multiple cells

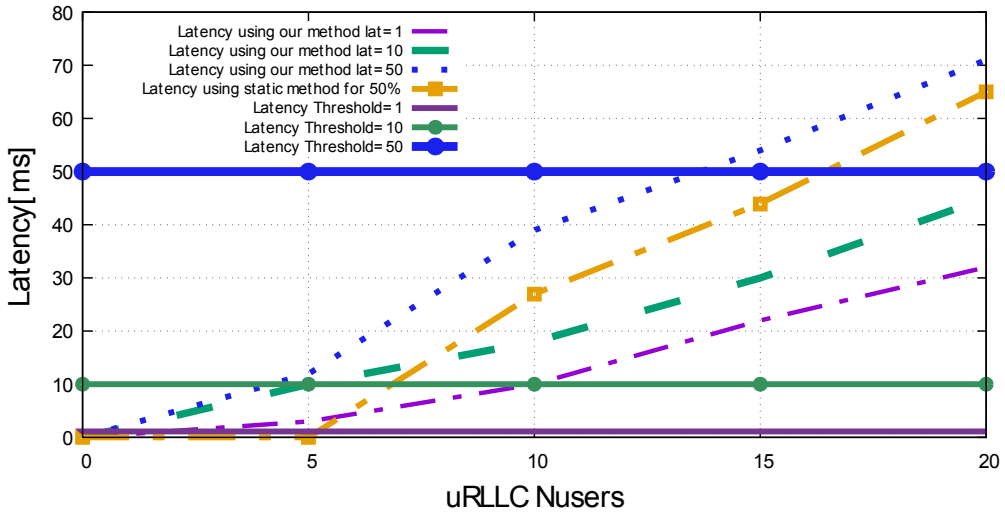
Result: $N_{pRBopt}(i,j)$
for *each cell* i **do**
  for *each slice* j **do**
    if Slice (j) == eMBB **then**
      $N_{pRBopt}(i,j) = \sum_{k=1}^{k=N_{users}(i,j)} \frac{d_{App/user}(j,k)}{d_{pRB}(j,k)}$
    else
      if Slice (j) == uRLLC **then**
        $N_{pRBopt}(i,j) = \frac{\mu_{opt} * avg\_packet\_size}{\frac{1}{N_{users}(i,j)} \sum_{k=1}^{k=N_{users}(i,j)} d_{pRB}(j,k)}$
      end if
    end if
    =0
  end
end

7 to 9; (ii) good quality where the CQI varies from 13 to 15. The different channel qualities will directly affect $dpRB$, which allows to see its impact on the proposed solutions. Note that we simulated the case of only one eNB and one SO. Table 3.1 presents the simulation parameter set in all scenarios:

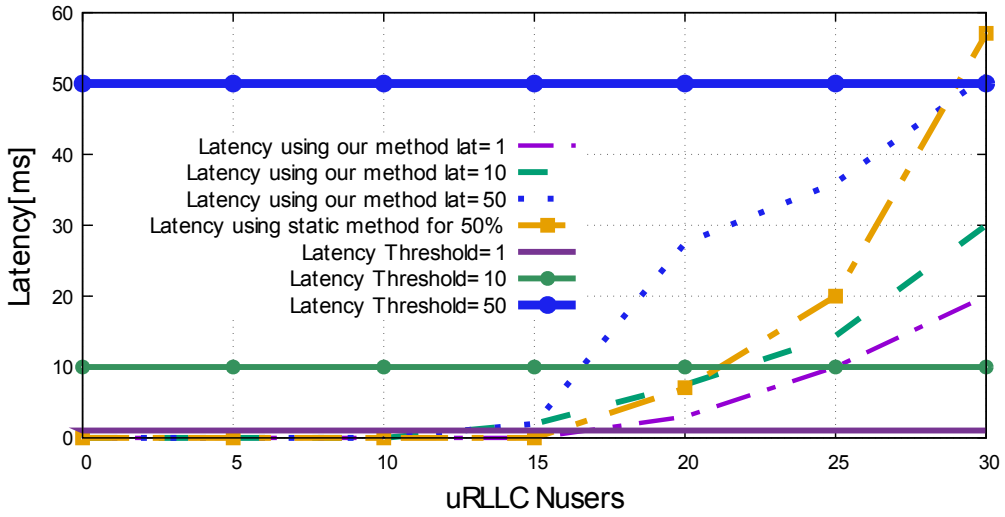
Table 3.1: Simulation parameters

Parameter	**Values**
Slices	[uRLLC, eMBB]
Average Packet Size	[20, 125] bytes
Data rate	[160, 1000] kbit/s
TTI	[1, 1] ms

We compared our solution with the one adopted in [12], which shares the $pRBs$ among the different slices using a statically selected percentage; in our tests, we considered a 50% slice-dedicated bandwidth (SDB) per slice. It is worth noting that the number of available $pRBs$ is bounded by the channel bandwidth. For our simulation, we used a channel bandwidth of 5Mhz, where 25 $pRBs$ are available. We selected this number to saturate the channel quickly and show the efficiency of our solution. For higher bandwidths, the only difference concerns the threshold from where our solution does not perform well. It may happen that the combined number of pRBs to be allocated to both eMBB and uRLLC exceeds the channel capacity; hence, we adopted in this implementation a fair



(a) Medium channel quality.

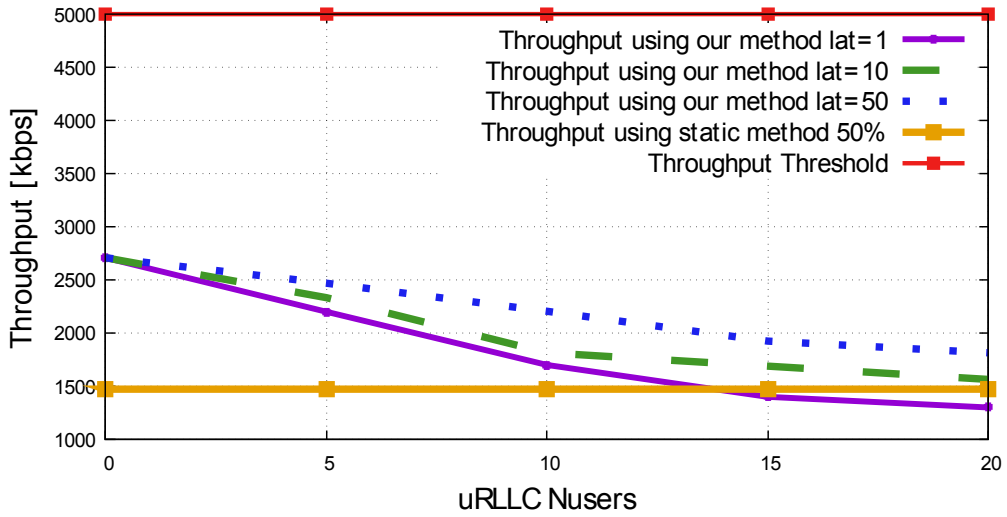


(b) Good channel quality.

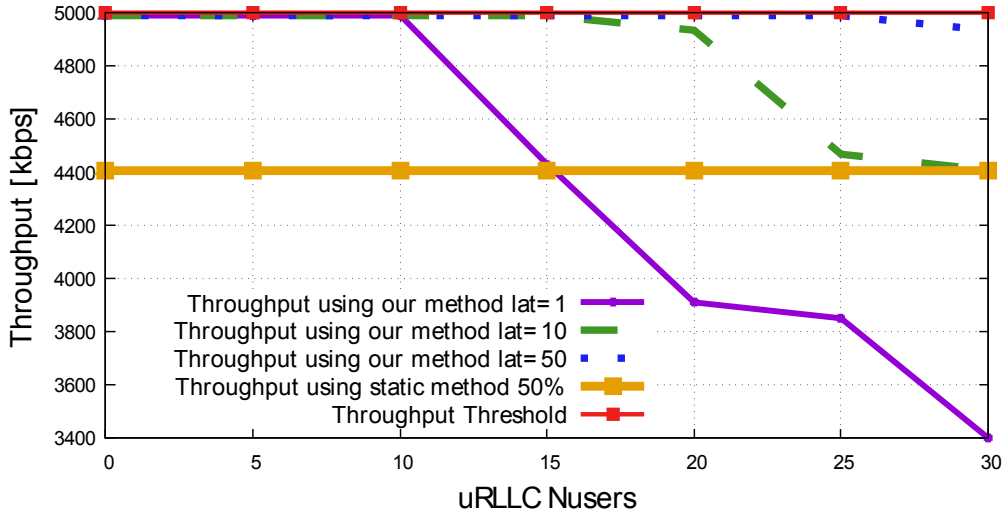
Figure 3.3: Latency vs. the number of uRLLC users.

share of the resources, which has been computed as follows. First, we compute $\Delta = N_{pRBmax} - (N_{pRBuRLLC} + N_{pRBeMBB})$ that represents the difference between the available number of pRBs and the requested number of pRBs for both slices. Then, we reduce the same amount of pRB ($\frac{|\Delta|}{2}$) from each slice in order to fit the capacity of the channel. Other policies could be used, such as giving high priority to one slice by first satisfying this slice and giving the remaining pRBs to the other slice. In this work, we use only the fair share of the channel, leaving other policies for future work.

Finally, we computed three main metrics: the eMBB slice throughput, the uRLLC latency and the variation of N_{pRB} for each slice. We varied the number of uRLLC slice users from 1 to 20 in the case of the medium-quality channel and from 1 to 30 in the case of the good-quality channel while fixing the number of eMBB users to



(a) Medium channel quality.



(b) Good channel quality.

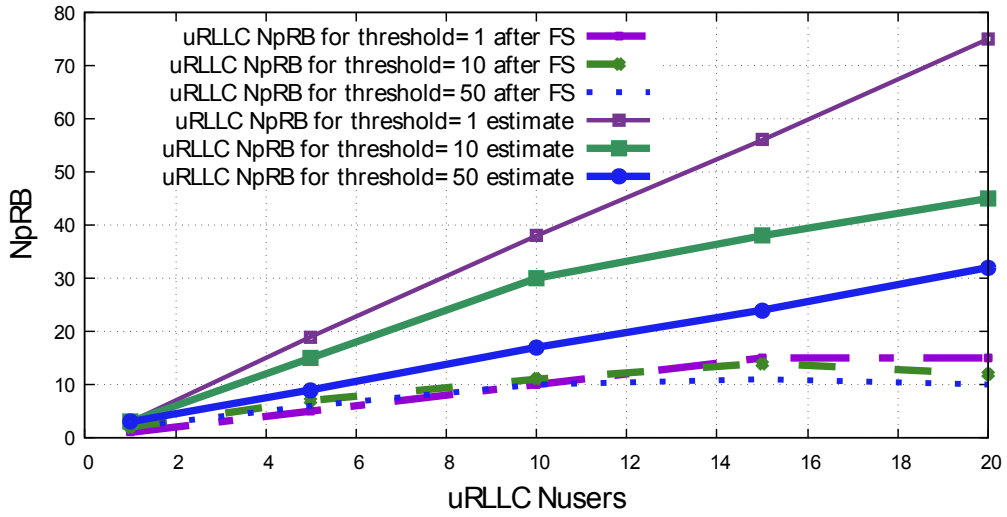
Figure 3.4: Throughput vs. the number of uRLLC users.

5. The presented results are averaged after several runs of the simulation.
It is worth noting that our main objective is to evaluate the accuracy of our proposed methods to well estimate the needed radio resources for each type of slice.

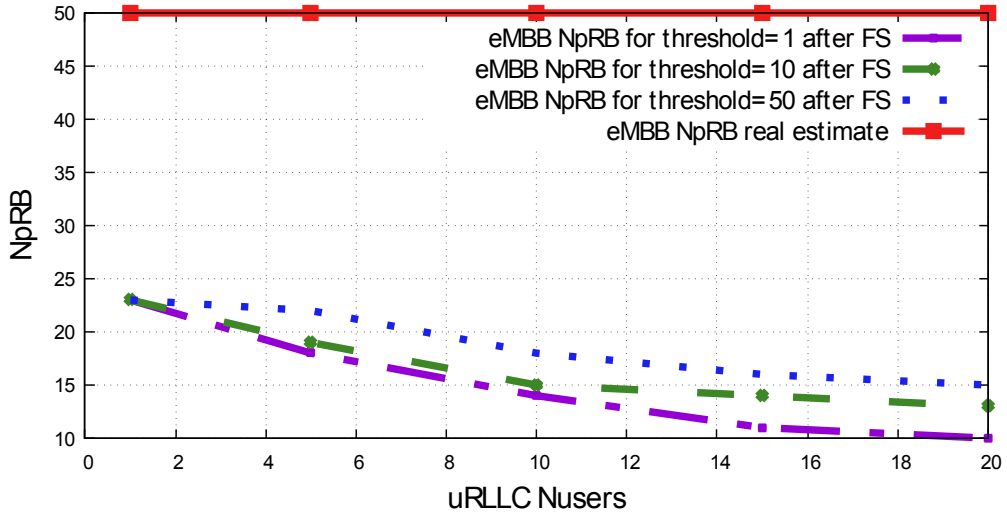
3.4.2 Results

Figure 3.3a and 3.3b illustrate the latency experienced by the uRLLC users for different numbers of N_{users_uRLLC}, and for two channel qualities, good and medium. Here we considered different values for Lat_{max}: 1 ms, 10 ms and 50 ms, which reflect different service-level requirements. We remark that our algorithm allows to keep the latency around Lat_{max}, whatever the value of the latter and for both channel qualities. However, we see that there is a threshold (i.e., number of uRLLC users)

beyond which latency exceeds Lat_{max}; 2, 5 and 14 for the medium channel quality for Lat_{max}=1 ms, 5 ms, and 50 ms respectively, and 15, 22 and 29 for the good channel quality for Lat_{max}=1 ms, 5 ms, and 50 ms respectively. The difference between these values is explained by the fact that good channel quality permits to have higher N_{pRB} compared with the medium channel quality, thus accommodating more uRLLC users. In addition, we observe that using a fixed number of pRBs cannot guarantee the very low latency requirement, as the used value (i.e., 50%) is not optimal (see Figure 3.3a and 3.3b).



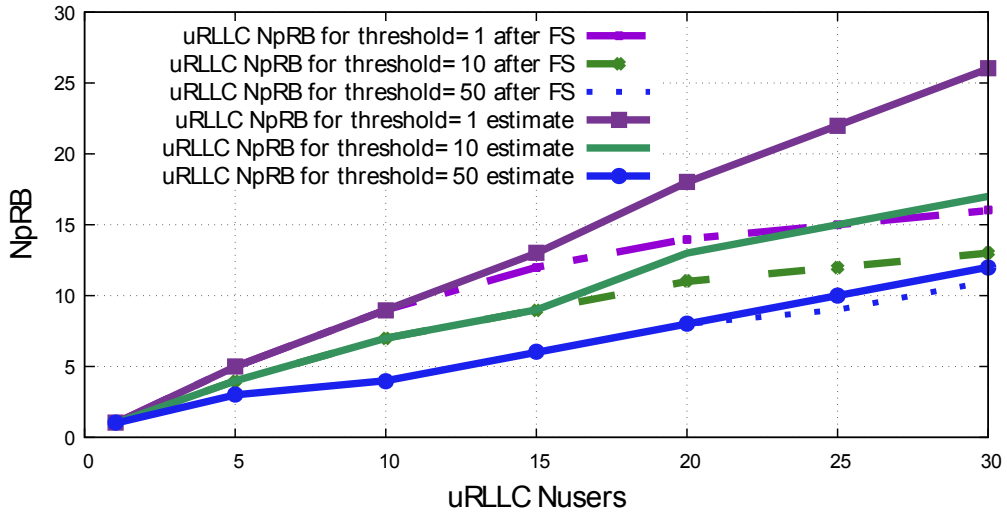
(a) N_{pRB} of uRLLC slice.



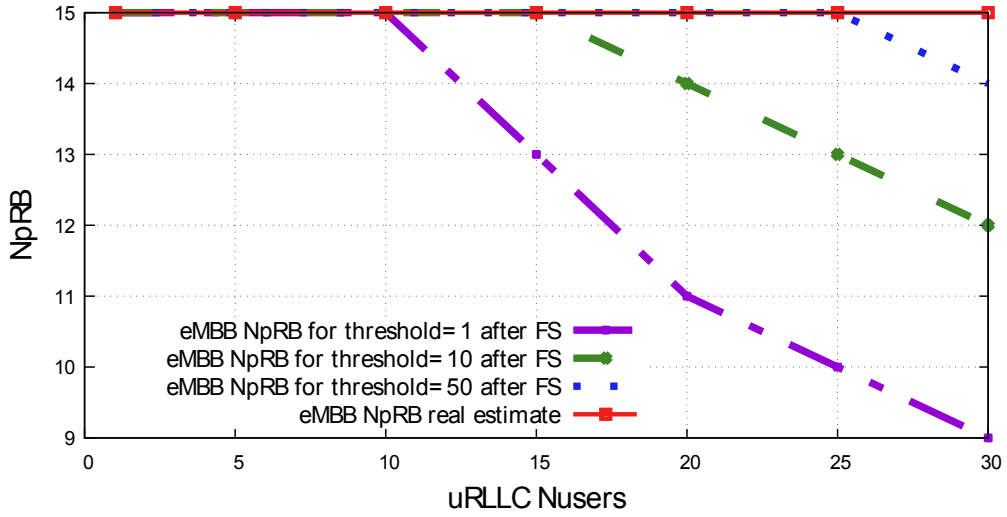
(b) N_{pRB} of eMBB slice.

Figure 3.5: Number of pRBs vs. the number of uRLLC users for a medium channel quality.

Figure 3.4a and 3.4b show the throughput obtained for the eMBB slice as a function of the number of the uRLLC slice's users. We remark the same behaviour as in the precedent figures. Namely, there is a threshold beyond which the performance



(a) N_{pRB} of uRLLC slice.



(b) N_{pRB} of eMBB slice.

Figure 3.6: Number of pRBs vs. the number of uRLLC users for a good channel quality.

of the slice degrades, particularly in the case of good channel quality. Indeed, for the medium channel quality, our solution cannot guarantee the requested bandwidth (5 users $\times$ 1 Mbps). However, for good channel quality our solution guarantees the needed bandwidth until 10 and 25 users when Lat_{max}=1 ms and 50 ms, respectively. This is expected, as in the case of Lat_{max}=1 ms the uRLLC users need more pRBs, which strongly affects the eMBB users (see Figures 3.5b and 3.6b). Regarding the static assignment of pRBs, it always ensures the same throughput (lower than 5mbps), which is not optimal.

To better understand the obtained results, we have drawn in Figure 3.5 and 3.6 the N_{pRB} estimated and used by the eNBs for each type of slice, and for both channel qualities. From Figure 3.5a and 3.6a we clearly see that the estimated

value of N_{pRB} is similar to the one communicated to the eNB, until reaching the identified thresholds in Figure 3.3a and 3.3b. When exceeding these thresholds, the communicated N_{pRB} to eNB are lower than the estimated value. This is mainly because the channel capacity is exceeded, and the proposed solution starts using the fair share of pRBs among the two slices. Hence, it is possible to accommodate the uRLLC requirement for both channel qualities.
Regarding the eMBB slice, where the results are displayed in Figure 3.5b and 3.6b for both channel qualities, the estimated N_{pRB} cannot be satisfied in case of medium channel quality, and after exceeding the identified threshold in Figure 3.4a and 3.4b for good channel quality. Furthermore, we remark that the number of needed N_{pRB} is higher in the case of medium channel quality, which is expected as $dpRB$ in this case is lower than when the channel quality is better; hence more pRBs are needed to satisfy the throughput of eMBB users.
Overall, these results confirm that the proposed model to estimate the needed N_{pRB} for eMBB and uRLLC employed by our proposed solution is accurate and permits to solve the problem of sharing the RAN resources among slices, as long as resources are available in the BW.

3.5 Conclusion

In this chapter, we addressed the problem of slicing and isolating RAN resources in slicing-ready 5G networks using the concept of two-level scheduling introduced in [12]. We proposed two algorithms that estimate the needed RAN resources for two types of 5G slices: eMBB and uRLLC. We used simulation to evaluate the performance of the proposed algorithms under different channel conditions. The obtained results allowed to verify the accuracy of our algorithms when estimating the needed pRBs for each type of slice, as long as resources are available in the BW. The proposed algorithms are used at the SO level and could be easily implemented in a real platform.
The algorithms proposed here for predicting the number of pRBs to be allocated to each slice are based on CQI feedbacks between the eNB and the SO. However, the frequent retrieval of these CQI values can saturate the eNB-SO links and negatively impact RAN performance. In order to solve this problem, we have proposed an algorithm to decrease the frequency of CQI recovery, which we will detail in the next chapter.

Chapter 4

Channel stability prediction to optimize signaling overhead in 5G networks using ML

Channel quality feedback is crucial for the operation of 4G and 5G radio networks, as it allows to control UE connectivity, transmission scheduling, and the modulation and rate of the data transmitted over the wireless link. In the previous chapter, we have used the CQI feedback between eNB and SO to update the N_{pRB} values dynamically. However, when such feedback is frequent, and the number of UEs in a cell is large, the eNB-SO links may be overloaded by signaling messages, resulting in lower throughput and data loss. Optimizing this signaling process thus represents a key challenge. In this chapter, we focus on CQI reports that are periodically sent from the eNB to the SO. In this context, we apply ML mechanisms to predict channel stability, which can be used to decide if the CQI of a UE is necessary to be reported, and in turn to control the reporting frequency. We study two ML models for this purpose, namely SVM and NN. Simulation results show that both provide a high prediction accuracy, with NN consistently outperforming SVM in our settings, especially as CQI reporting frequency reduces.

4.1 Context and motivations

Providing reliable communication technology represents a key challenge for 5G systems in both CN and RAN levels. In order to achieve reliability at the RAN level, an eNB or gNB (in 4G and 5G terminology, respectively) should allocate a sufficient amount of radio resources per UE and appropriately select the MCS in order to meet the requirements of each considered application. The amount and configuration of these resources, i.e., pRB, are directly related with the channel conditions at the UE end. For this reason, the eNB should ideally know in real-time the quality of

the channel of each device, which allows it to properly schedule the necessary N_{pRB} for transmission [83]. In 4G and 5G networks, this number depends on the CQI value, which is periodically reported by UEs to the eNB/gNB, and conveys their current communication channel quality [84]. Nevertheless, the periodic transmission of CQI information incurs signaling overhead; this may overload eNB-SO links and negatively impact RAN performance. Therefore, it is important to optimize this signaling process in order to be able to improve on QoS.
This work is put in the context of the 5G RAN slicing design presented in [12], including a SO responsible for cross-slice resource sharing [18]. This architecture requires accurate channel quality information per UE at the gNB and at the SO level in order to be able to estimate and dynamically adjust the radio resource allocation to satisfy the heterogeneous requirements of coexisting network slices. This information is reported by UEs via standard procedures and propagates to the SO via a southbound protocol by base stations. The challenge that we face and the particular motivation here is to reduce this reporting overhead.
The key elements responsible for fluctuations in CQI values are the radio environment changes, which may be due to user mobility, multi-path effects, and other phenomena. We introduce the term *channel mobility* to denote time-varying changes in the radio environment of a UE: On the one hand, the channel is considered static if its conditions are mostly stable when typically the UE is static or low-mobility for a period of time. Thus, the reported CQI values remain constant or show minimal variation, which does not impact radio resource allocation. On the other hand, the channel is considered mobile when it varies significantly due to factors such as UE mobility and other effects. In this case, the CQI values exhibit significant fluctuations. Consequently, it is crucial that the base station is informed about the changed channel quality information in order to determine the appropriate amount of resources to be allocated and update the N_{pRB} values for the different UEs.

Our contribution here is in the direction of reducing the signaling overhead by optimizing the reporting of CQI information via limiting the amount of unnecessary transmitted messages, at the same time ensuring that the SO has an accurate view of the SO-eNB link conditions. Detecting whether the channel is static or mobile over time, though, is challenging, and this is the main issue we address in this chapter. To this end, we apply ML techniques in order to be able to detect channel variations. Our approach involves collecting data from the system in order to study, analyze, and extract the information needed to make a decision. In fact, having a lot of different types of data about per-UE channel quality, complemented with other information such as user mobility patterns, fine-grained geographical locations, etc. would assist in getting a more accurate view from the output of the ML algorithm to properly identify the stability of the channel. However, this may not be feasible for technical and privacy reasons.

In this chapter, we make the following contributions: first, we propose a ML-

driven methodology to predict channel mobility and accordingly adapt the CQI reporting frequency, aiming to reduce signaling overhead while maintaining an accurate view of channel conditions per UE to appropriately allocate radio resources. We study, analyze, and predict the channel's state using two different ML algorithms to evaluate their suitability and select the more accurate one for our purposes. Our first design goal is to avoid collecting many data metrics, thus focusing only on the CQI parameter; the features we have selected for training and classification can be calculated solely by the statistical processing of the collected CQI values. Our second design goal is to avoid collecting large volumes of CQI data; to this end, we evaluate different frequencies to collect these data and their impact on the accuracy of identifying channel mobility.

4.2 Optimization of CQI signaling overhead state of the art

Several research works in the literature have been elaborated in order to control the transmission of CQI information, allowing the optimization of the signaling overhead. Two families of such control techniques have been devised.

4.2.1 Techniques based on frequency

Techniques of this kind are based on compression models. Indeed, the idea to reduce the signaling overhead consists of sending a compressed CQI value of a series of pRBs, instead of sending a CQI value for each one. In this context, three categories were proposed as follows [85]: i) Broadband compression, where a single CQI value transmitted refers to all pRBs of the bandwidth, ii) sub-band compression, where the bandwidth is divided into multiple sub-bands with the same size, and the UE selects only one CQI value to be transmitted to the base station, and iii) full band compression, where the base station estimates the total bandwidth quality, using mathematical transformations such as the discrete cosine transform and the Haar wavelet transform. Sivridis and He [86] presented a non-predictive signaling reduction scheme, where users with a high signal-to-interference-plus-noise ratio (SINR) transmit only broadband information, while users with low SINR are allowed to return on-demand instant CQI information at high rates. Therefore, a technique was proposed to determine the threshold that separates users required to use full-band feedback from users required to use compression in the wide-band frequency domain. The work of Kang and Kim [87] is based on the sub-band compression method, allowing to analyze and select the best M-feedback for OFDMA systems. In addition, a combined optimization was applied to minimize overhead feedback costs based on the number of reported RBs per user and the signal to quantization noise ratio bits. Abdulhasan et al. [88] presented a compression scheme for CQIs in

a 3GPP-LTE and LTE-A system, where CQI values are communicated to eNodeBs based on a defined threshold. A trade-off was presented to select the appropriate threshold since a high threshold is recommended for high-speed conditions, whereas a low threshold is recommended to ensure reliable transmission mainly in an overloaded network.

4.2.2 Techniques based on timing

Chiumento et al. [89] proposed an estimation method of the channel quality based on Gaussian Process regression at the base station. This is achieved thanks to an adaptive and online CQI prediction scheme allowing estimation of the channel quality variations and the behavior monitoring of each user. In [90], the authors dealt with the CQI aging problem, which could be defined by the mismatch between the CQI used for the channel adaptation and the current state of the channel. This problem is caused due to processing delays or because of infrequent CQI reporting. To overcome this problem, the authors proposed a comparison study of various signal-to-noise ratio prediction algorithms, such as Kalman filters, among others. In [91], an algorithm is presented for dynamic CQI resource allocation using ARQ information, Doppler mobility monitoring, and the MAC layer service classifier. The idea consists in the combination of information from the physical and MAC layers, providing additional information about the channel quality and its effect on MAC frames in terms of delay, packet error rate, etc. This approach allows tuning the periodicity of the feedback window in order to address QoS, robustness, and feedback overhead tradeoffs.
The proposed methods to optimize signaling overhead provided relatively interesting results. However, almost all of them consisted of the prediction of channel quality based on complex optimization algorithms, as well as required several input parameters that are often not feasible to acquire nor accurate in some conditions. Our approach is inspired by these schemes, but it is applied for CQI feedback between e/gNB and SO, and it is based on a simpler intelligent mechanism, which only requires as input CQI information for accurate channel mobility/stability prediction.

4.3 Channel stability prediction using machine learning

This section focuses on the description of the proposed concept to predict channel stability based on ML. This mechanism can then be used to optimize the transmission of CQI data messages and thus reduce the associated signaling overhead.

4.3.1 Overview and objectives

CQI messages are sent periodically from e/gNB to the SO, in order to provide it information about the channel quality allowing it to appropriately allocate resources. When channel quality is relatively stable, the CQI values do not vary a lot. Therefore, an increased CQI reporting frequency does not contribute to the view the base station has on the actual radio conditions of a UE link and does not affect the quality of the radio resource allocation. We thus take advantage of channel stability to avoid transmitting unnecessary CQI reports and alleviate the associated overhead. Our approach consists of monitoring the channel state for a period T. If channel mobility is identified by the predictor, a new CQI value is required to adjust resource allocation. Otherwise, there is no need to receive new CQI values; the SO allocates radio resources considering the last received CQI value as accurate and stable, and the CQI reporting frequency can be reduced. The different steps of this concept are illustrated in Figure 4.1.

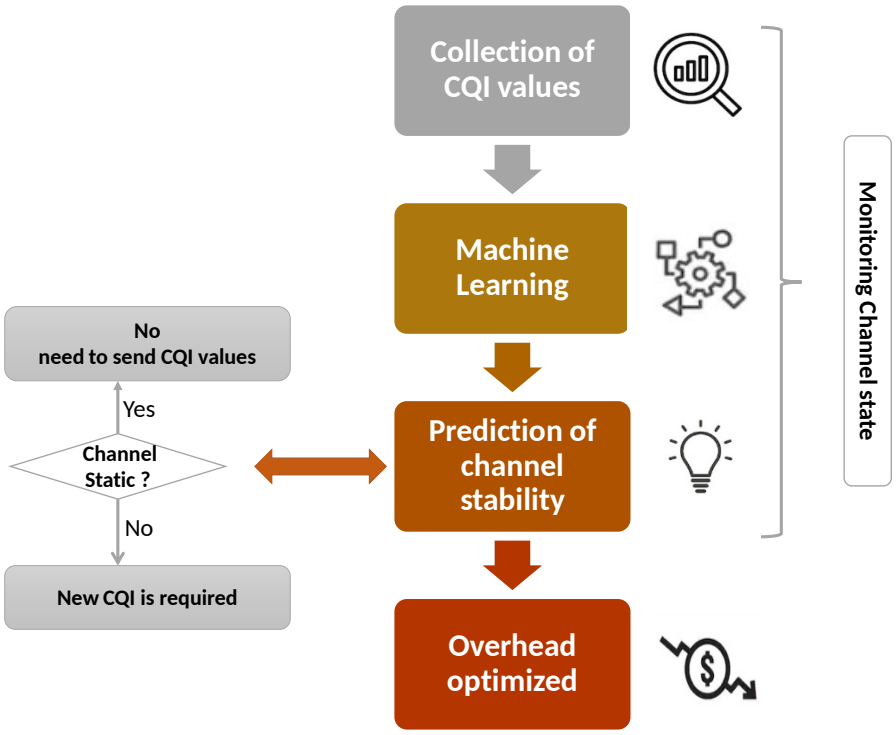


Figure 4.1: Our concept and methodology to reduce CQI monitoring overhead.

The proposed monitoring phase is based on a ML algorithm, which helps predict the channel state. In this contribution, we have tested some ML algorithms, then we have selected SVM and NN as they offer better accuracy.The next section focuses on describing how these two ML schemes are applied in our network settings in order to predict channel mobility based on different frequencies of collected data.

4.3.2 Steps to predict the channel's state

In this section, we present in detail the steps involved in our ML-based methodology and the metrics we use to evaluate the performance of the candidate algorithms for channel mobility prediction. Indeed, there are different ML approaches [92]. The most commonly used are: i) Supervised learning, which consists of training the algorithm using a set of data consisting of an input and a desired output. Then, a function that maps an input to output is inferred based on the training data. This function allows the mapping of new unseen data instances to output values, which may correspond to distinct classes; ii) Unsupervised learning, which consists of learning in a self-organized way allowing to find an unknown sample from a set of data without being based on an existing label; and iii) Reinforcement learning, which allows to decision making by interacting with an environment, formulated as a Markov decision process. It has similarities with supervised learning but without the need for labeled input/output pairs.
In this work, we apply supervised learning techniques and define two classes (static and mobile) based on the CQI parameter. We evaluate two supervised learning algorithms in order to predict the channel state and assign it to the appropriate class:

- NN [93]: This mechanism is modeled and inspired by the human brain, aiming to create an artificial neural network. The concept consists of learning the machine by incorporating new data. The machine typically consists of different layers of interconnected neurons, each one of which interprets the input data through a kind of machine perception and sends an output to a connected neuron until the last layer provides the output of the system.

- SVM [94]: This approach consists of learning from a set of multi-dimensional data vectors labeled by their category (class) and creating a model used to classify new data by finding the hyperplane that separates the training data by the optimal (maximum) margin. SVM is a binary *linear* classifier.

Figure 4.2 presents the different steps involved in the channel state prediction process.

A. Feature vector creation and labeling phase

This phase involves the collection of data and their processing in order to extract specific features and create feature vectors (also called characteristic vectors) that will be used for training a classifier. Raw data are collected in the form of vectors for different channels during a period T. A feature vector is then created for each data vector (i.e., for each channel).
In fact, different types of data representing the channel state may be used, such as SNIR, CQI, and others [95]. We select the CQI parameter for the proposed predictive system, as this parameter provides sufficient information on the channel state

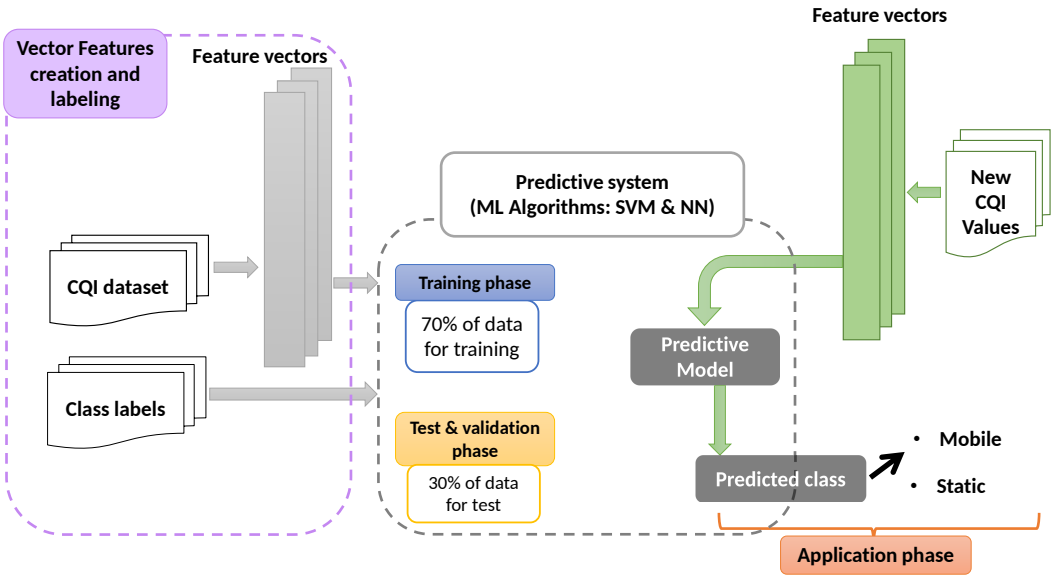


Figure 4.2: ML-driven channel stability prediction.

and is used by the MAC scheduler to allocate resources and decide on parameters such as the MCS. We extract a feature vector from each CQI data vector after a preprocessing step in order to have the relevant data for the predictive system to identify the channel state (mobile or static).
Preprocessing is carried out on the data vector $CQI_T = [cqi_1, cqi_2....cqi_n]$ of n CQI values collected during a period T in order to extract the characteristic vector C= $[C_1\ C_2\ C_3]$. The extracted features are the following.

- C_1: The difference between the maximum and minimum values of collected CQIs in the data vector CQI_T.

$$C_1 = cqi_{max} - cqi_{min} \tag{4.1}$$

  The channel may be static if C_1 is small or zero, which might mean that the UE is static, and the environment is stable (there are no significant effects that cause a drastic change in the CQI value). This feature can provide an idea of the channel state, but it is not sufficient to make a decision.

- C_2: Variance.

$$C_2 = \frac{1}{n}\sum_{n=1}^{n}(cqi_i - \overline{CQI_T}) \tag{4.2}$$

  This feature measures the dispersion of CQI values relatively to the average $\overline{CQI_T}$, which characterizes the level at which the CQI can have a value more or less far from its expectation.

- C_3: The vertical change of the CQI curve slope, representing the CQI change in different samples in period T.

$$C_3 =\mid CQI(t_{i+\Delta}) - CQI(t_i) \mid, \tag{4.3}$$

  where $CQI(t_i)$ and $CQI(t_{i+\Delta})$ are the CQIs collected at t_i and $t_{i+\Delta}$ respectively, where these two times are inside the sample (Δ=5 in our case). Multiple C_3 values are extracted for each sample. Thus, the size of C depends on the number of C_3.

After the creation of vector C, a known label (static or mobile) is assigned to it, in order to be used for the training phase.

B. Creation of a ML-based predictive system

To create the predictive system, a ML algorithm operates in two phases as follows.

Training phase 70% of the feature vectors with their labels (representing the real classes) are used to train the classifier. During this training phase, the ML algorithm creates a function that maps inputs (feature vectors) to outputs (labels), used then to classify new vectors. In this stage, the SVM algorithm learns a linear function, while the NN algorithm also supports nonlinear functions.

Test and validation phase This phase uses the rest of the feature vectors (30%). It consists of checking the predicted classes of these vectors against their assigned labels. The validation of the predictive system is based on a confusion matrix [96], which consists of the number correctly and incorrectly classified samples per class. Performance is evaluated in terms of the following metrics:

- Accuracy, i.e., the ratio of the number of correctly predicted vectors to the total number of vectors.

$$\text{Accuracy} = \frac{\#\text{ correctly predicted}}{\#\text{ feature vectors}} \tag{4.4}$$

- F1-score, which is defined by the weighted average of *precision* and *recall*, where precision is the ratio of the number of correctly predicted mobile class instances to the total number of predicted mobile class ones (i.e., false and correct), and recall, also called sensitivity, is the ratio of the number of correctly predicted instances of the mobile class to the number of all true mobile class ones.

$$\text{F1.score} = \frac{2 * (\text{Recall} * \text{Precision})}{(\text{Recall} + \text{Precision})} \tag{4.5}$$

Application phase

This step consists in classifying a new CQI data set over different frequencies of collecting data (i.e., sample size variation). To evaluate this phase, we use the *True Positive Rate (TPR)* and the *True Negative Rate (TNR)* metrics, where the positive class refers to the mobile class and the negative one refers to the static class. TPR and TNR are defined as follows:

$$TPR = \frac{\#\text{ correctly classified as mobile}}{\#\text{ mobile}} \tag{4.6}$$

$$TNR = \frac{\#\text{ correctly classified as static}}{\#\text{ static}} \tag{4.7}$$

4.4 Performance evaluation

This section focuses on the performance evaluation of the channel mobility predictive system provided by the two ML algorithms (NN and SVM). We first created the dataset by generating CQI values for different channel mobility states using the `ns-3` simulator. Then, we used MATLAB [97] [98] to train and test ML algorithms based on the provided data set, as well as to evaluate new CQI data sets with different CQI collection frequencies. For the NN case, we trained a neural network with a single hidden layer using the Levenberg-Marquardt algorithm. We experimented with different layer sizes and found that using 10 neurons in the hidden layer provided the best accuracy among the options that we tested. Performance is evaluated first for the test and validation phase, and then for the application phase for both ML algorithms.

4.4.1 Test and validation phase evaluation

In order to create a data set with realistic CQI values corresponding to different degrees of user mobility, we simulated an LTE cell using `ns-3`, where UEs move with different constant velocities. We thus generated approximately $15,500$ vectors of CQI values with different channel mobility states and extracted a feature vector for each CQI vector as described in Section 4.3.2, which we labelled either as static or mobile, depending on the level of UE mobility. Note that a feature vector is calculated on a sequence of CQI values collected during a time period $T = 400\,\text{ms}$. For both ML algorithms considered, we use 70% of our data for training and the remaining 30% for test and validation. Table 4.1 presents the results of the validation phase in terms of accuracy and F1-score for the two candidate ML mechanisms.

As shown in this table, both algorithms are able to learn and predict the channel state with high performance, as they provide accuracy and F1-score of more than 90%. We notice, though that the NN scheme outperforms SVM in terms of accuracy and F1-score by approximately 4%.

Table 4.1: Accuracy and F1-score of NN and SVM algorithm

	NN	**SVM**
Accuracy	96.43%	92.86%
F1-score	96.29%	92.30%

This can be explained by the fact that SVM is based on the margin maximization of the linear hyperplane separator between the two classes (static and mobile) [94]. That is why it is not able to well classify vectors close to this separation. Contrariwise, NN is based on a non-linear function to separate between classes allowing it to better handle such cases.

4.4.2 Application phase evaluation

After the test and validation of the predictive system, we use `ns-3` simulations in a similar way to create a new CQI dataset for the application phase in order to evaluate the efficiency of our classifiers and evaluate their behavior for different CQI collection frequencies. The generated feature vectors are created from raw data that correspond to two types of UE mobility (mobile and stable) and are labeled as such. Four groups of test data were generated, each for a different CQI reporting frequency, namely every 2, 10, 50, and 100 ms. Performance is evaluated in terms of TPR for the mobile class and TNR for the static class. The obtained results are as follows:

- The static channel state obtains TNR performance between 99% and 100% for any CQI collection frequency and for both NN and SVM algorithms. In fact, for this class, all CQI values are relatively close to each other. This is why the selected samples with the different frequencies provide a small variation in the features of the vector F, allowing to identify the channel as static. Therefore, it is evident that both algorithms succeed in correctly predicting channel mobility in the static case.

- The prediction of the mobile channel state is harder, as CQI values are highly varied. Therefore, the predictive system should detect the variation of the CQI values with different data collection frequencies and appropriately select samples (with the varied CQI values) on which it is based to select the appropriate class. The obtained results of TPR performance for the different reporting periods are illustrated in Figure 4.3.

  As shown in this figure, for small periods ($2ms$ and $10ms$), both algorithms achieve a high TPR performance (more than 95%). It can also be noticed that there is a small increase with the NN algorithm. However, for low CQI

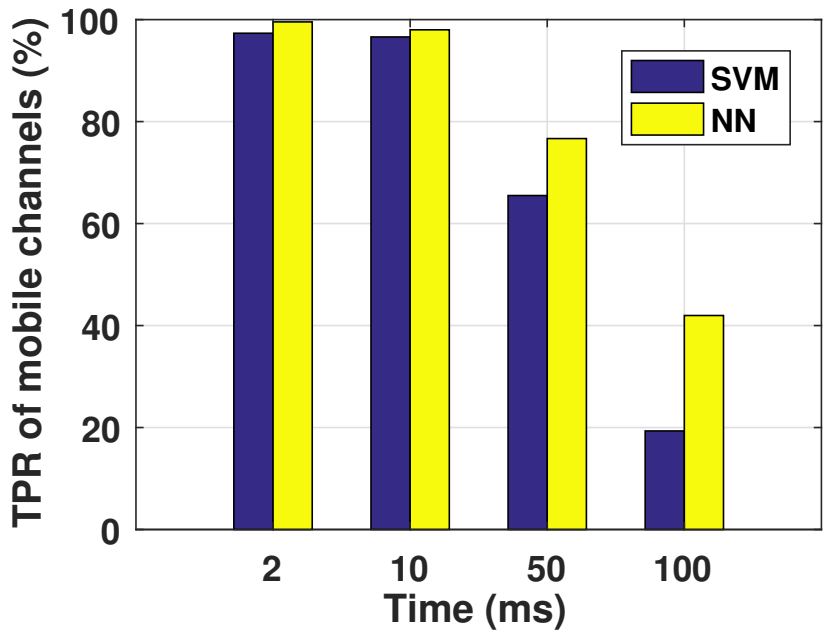
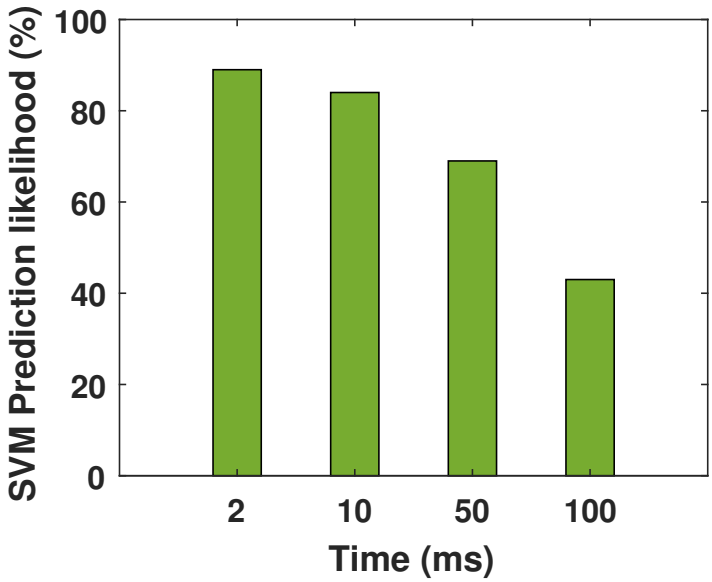


Figure 4.3: TPR of mobile channel for different CQI reporting frequencies. The x-axis represents the period between two consecutive CQI reports by a UE.

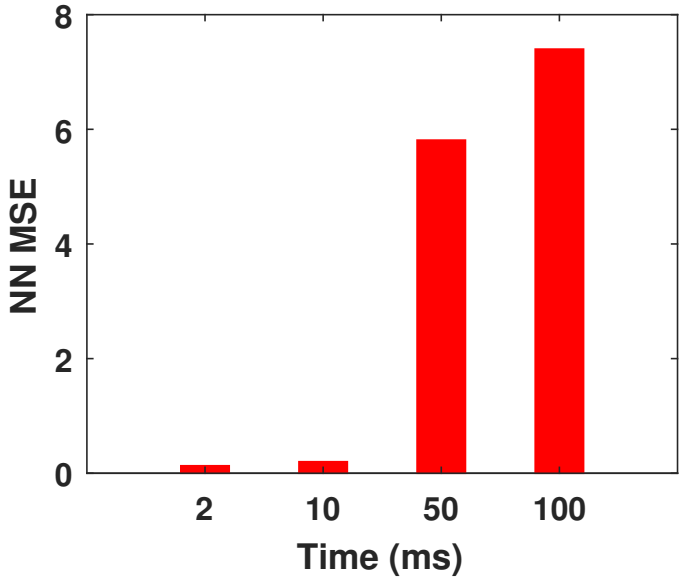
collection frequencies, TPR is significantly reduced for both NN and SVM algorithms. For $50ms$ and $100ms$, TPR is respectively around 76.67% and 42% with NN, compared to 65.5% and 19% respectively when using SVM. These results show that the predictive system provided by NN has the ability to detect channel mobility more effectively and is more robust when CQI is reported with a lower frequency (large periods). Although when the reporting period is $100ms$, the TPR is relatively low for NN, it still achieves a $2\times$ increase in performance compared to the TPR of the SVM algorithm. The NN algorithm outperforms SVM, thanks to the non-linear function used to separate between classes.

4.4.3 Prediction quality evaluation

This part focuses on evaluating the quality of the prediction of the NN and SVM-based approaches relatively to different CQI collection frequencies. To quantify it, we rely on appropriate prediction quality metrics for each ML scheme. In particular, the metric we use for SVM is the prediction likelihood (also called prediction probability), which is an expression of the certainty that a feature vector is correctly classified. It is calculated by taking into account how far the score returned by the SVM classifier is from the threshold value that classifies a test vector. In a similar sense, the metric we use for the NN classifier is the Mean Squared Error (MSE), which conveys the uncertainty about the correctness of the classification.



(a) Efficiency of SVM



(b) MSE of NN

Figure 4.4: Prediction score for different CQI reporting frequencies. The x-axis represents the period between two consecutive CQI reports by a UE.

As illustrated in Figure 4.4, when the CQI collection frequency decreases, the likelihood to correctly predict the channel state by the SVM algorithm reduces and the MSE of the NN algorithm increases. These results are due to the fact that when the CQI reporting frequency is smaller, there are fewer raw CQI samples during the time window $T = 400\,\text{ms}$ out of which a feature vector is created. This lost information has often the effect that the variability of CQI values in a window

decreases. There are cases when the real value of the CQI between consecutive samples in a window fluctuates, but this is not captured in the data samples, making the CQI appear to remain mostly constant during the collection period and, in turn, causing the algorithm to misclassify the channel as static. For these reasons, the prediction error rate impacts the TPR as presented in Figure 4.3, where the TPR drops as the collection data frequency decreases.

4.5 Conclusion

In this chapter, we focused on ways to reduce the signaling overhead caused by the periodic transmission of channel quality feedback in the form of CQI reports between the e/gNB and the SO in 4G and 5G mobile networks. Our approach consists of avoiding to transmit unnecessary CQI messages by taking into account the stability of channel conditions, i.e., reducing the amount of CQI reports when the value of the latter does not change significantly over time, as a result of a stable channel. To this end, we addressed the challenge of predicting the channel's stability, proposing ML-based mechanisms that only require CQI information as input. Our mechanisms thus operate in a standards-compliant way and require no cross-layer or other external information, such as user locations or mobility patterns. We compared two ML schemes for this purpose, namely SVM and NN, evaluating and analyzing their prediction accuracy. We further addressed the tradeoff between prediction accuracy and data collection frequency and experimentally showed neural networks to consistently outperform SVMs in all our settings.

In this chapter, we mainly focused on evaluating the prediction accuracy of the candidate ML schemes. The next chapter will launch a deeper study on the impact of our proposed methodology and mechanisms, integrating them in the 5G network slice management architecture that we have proposed in our prior work. Our immediate goal is to evaluate the signaling cost improvements that can be achieved, the impact of our proposed mechanisms on the allocated resources, and the attained performance in terms of latency and throughput for heterogeneous 5G network slices.

Chapter 5

Data-Driven RAN Slicing Mechanisms for 5G and Beyond

In Chapter 3, we have highlighted the issue of sharing resources between deployed slices in the RAN. Hence, we have presented algorithms at the SO level to drive the N_{pRB} to be used by each deployed network slice. These resources are adjusted periodically based on current estimates of achievable throughput performance derived from channel quality information, particularly from the CQI values of the users of each network slice retrieved from the RAN. CQI information is reported to base stations by the UE following standard procedures, but extracting and frequently reporting it from base stations to the SO may significantly increase communication overhead. To this end, in Chapter 4, we have introduced a ML approach to infer the stability of UE channel conditions, aiming to avoid sending CQI reports when their values do not change.
To mitigate this overhead while maintaining at the SO level an accurate view of UE channel qualities, in this Chapter we: (i) apply the method of channel stability used in Chapter 4, (ii) propose a predictive scheme to reduce the CQI reporting intensity based on the inferred channel status. The proposed methods are integrated with the resource sharing algorithms proposed in Chapter 3 in order to demonstrate the efficiency of a data-driven RAN slicing framework.

5.1 Context and motivations

Managing, isolating, and slicing network resources for the RAN becomes an increasingly difficult task that must be properly designed in order to improve network performance, to introduce flexibility, and to achieve greater utilization of network resources by providing only the necessary network resources to meet the requirements of the activated slices in the network. Each type of slice has its own characteristics and requirements in terms of latency and/or throughput. For example, eMBB

requires high bandwidth, while URLLC aims to minimize latency and maximize reliability.
Our contributions in Chapter 3 have addressed this problematic, by proposing resource sharing algorithms to compute and dynamically adjust the required radio resources to be used by each deployed network slice, covering the eMBB and URLLC slices. In this context, we first propose an algorithm that derives the number of pRBs needed by uRLLC and eMBB slices, according to their requirements in terms of latency and throughput, respectively. Then we design a RAN-aware dynamic slicing algorithm that exploits per-user RAN-level information to more accurately translate the derived service rate to an appropriate pRB assignment, using the CQI reports periodically obtained from eNBs. However, the high frequency of this CQI feedback overloads the channel with signaling messages, resulting in lower throughput and loss of data. Hence, the optimization of this signaling process represents a major challenge that we address in Chapter 4 by devising a mechanism to optimize the reporting process of CQI, which is periodically sent from the base station to the SO. The objective is to reduce the signaling overhead and avoid the associated channel overloads while maintaining at the SO level an accurate view of UE channel qualities. Indeed, we apply ML mechanisms to predict channel stability in a considered period t, which can be used to decide if the CQI is necessary to be reported in the next considered period $t+1$, and in turn, to control the reporting frequency.

In this chapter, we take advantage of channel stability study in period t to estimate the exchange frequency of CQIs over the next period $t+1$. Therefore, if at period t the channel is stable, there is no need to collect CQI in the next period $t+1$. However, if the channel is mobile we propose two techniques, namely the minimum difference and LSTM to estimate the CQI feedbacks frequency over the next period $t+1$.

5.2 RAN Slicing framework

In this chapter, we adopt the same architecture and assumption provided in Chapter 3, to estimate the N_{pRB} to allocate to each network slice, namely eMBB and uRLLC. In this context, the calculation of the initial N_{pRB} per slice (URLLC and eMBB) is based on the assumption that d_{pRB} is fixed for all users. However, users have different channel conditions, hence different data rates. In order to correct the estimation of the d_{pRB}, we propose to use information from per-UE channel quality reports obtained from e/gNBs. In 4G and 5G, the CQI reports are transmitted from UEs to eNBs and e/gNBs, respectively, via a standard procedure in order to be used in the scheduling process, and they include the CQI and MCS values of each UE belonging to a cell. This information should be made available to the SO and be used as input to our algorithms. Based on the CQI, we estimate d_{pRB} per UE and per cell (e/gNB). Indeed, d_{pRB} can be obtained based on the same tables

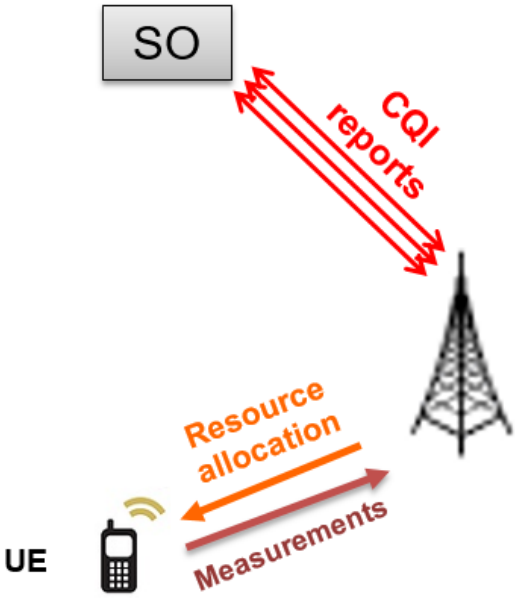


Figure 5.1: CQI reports exchange.

used by e/gNB to translate a CQI to a data rate [82] (translation table, refer to section 2.1.1).
Note that this algorithm is run by the SO periodically and relies on periodically transmitted CQI reports from e/gNBs. The periodicity of applying these algorithms is independent of the scheduling period TTI used at the MAC layer of e/gNBs.
The previously proposed algorithm is based on the periodic CQI reports sent by e/gNBs to the SO (refer to Figure 5.1), in order to correctly update the d_{pRB}. However, this CQI reporting may involve significant traffic overhead and can become an issue with a high number of slices or UEs. Hence, optimizing this signaling process is crucial and represents a challenge that we are addressing in the next sections.

5.3 Algorithms for reducing the CQI reporting frequency

As indicated earlier, the main issue of the precedent algorithm is the potential traffic overload due to frequent signaling messages (CQI reports between e/gNBs and the SO). Therefore, a key challenge consists in proposing mechanisms to optimize this procedure, reducing the frequency of CQI report transmissions. The idea focuses on limiting the amount of CQI reports and avoiding unnecessarily transmitted ones while ensuring that the SO maintains an accurate view of the state of the channel. In this context, Chapter 4 have presented the details of applying ML mechanisms

to predict channel stability/mobility, as this can be used to decide if a CQI value is necessary to be reported, and in turn, to control the reporting frequency. For instance, if the channel is stable, it is not necessary to frequently retrieve the CQI report since it does not vary during this time period.
Besides proposing the algorithms for channel stability analysis, we introduce in this section two algorithms: the first one allows to estimate the appropriate frequency of reports while minimizing the d_{pRB} estimation error as much as possible, i.e. by reducing the CQI computation frequency; the second is based on a ML-based prediction method, namely Long Short-Term Memory (LSTM), and has the purpose of forecasting a sequence of CQI or d_{pRB} values during a time interval based on past CQI or d_{pRB} values, respectively, without the need to retrieve the actual ones from the e/gNB. These two methods aim to reduce the number of CQI report exchanges, noted N_r, and expressed as follows:

$$N_r = \frac{T}{f_{rep}}, \tag{5.1}$$

where T is the duration of the test time period, and f_{rep} is the time interval between two successive CQI reports. This means that reducing N_r is equivalent to increasing f_{rep}.
Our objective here consists of reducing N_r (or increasing f_{rep}) of CQI report message exchanges as much as possible while reducing the error between the real d_{pRB} (using CQI reports) and the predicted one (when CQI is predicted).
Our system alternates between *monitoring* and *prediction* periods. For this purpose, we consider a (fixed-duration) time interval (τ) when a monitoring phase takes place by collecting a fixed number of CQI samples to evaluate the stability of the channel. On the one hand, if the channel is stable, there is no need to recover the CQI report for the next interval $\tau + 1$, which helps reduce the frequency of the CQI report exchange. On the other hand, if the channel is classified as mobile, we apply one of the proposed methods to either (i) predict the optimal number of CQI reports required for the next interval of equal duration or (ii) forecast a sequence of CQI values without actually retrieving them. These two methods are called *Optimal Difference* and *LSTM*, respectively, and are described in the following.
It is worth noting that for LSTM, the duration of each prediction period depends on the system QoS required. For example, the operator may decide to shorten a prediction period to sacrifice monitoring load gains in order to achieve a more accurate view of the actual UE channel conditions.

5.3.1 Optimal Difference method

This method is based on estimating the stability of the channel over a period τ and then selecting the appropriate number of CQI report exchanges N_r for the next period $\tau + 1$, while minimizing the error (E) between real and estimated d_{pRB}

corresponding to actual and predicted CQI values.
The challenge here consists in minimizing N_r and E, which cannot be solved by an optimization algorithm, as we do not have exact constraints on the error. Hence, we need to find a relation between E and N_r that allows defining the optimal N_r and E. Obviously, if N_r decreases (f_{rep} increases), the error E either remains the same or increases. Indeed, when N_r decreases, the frequency update of the d_{pRB} values decreases and consequently causes errors mainly for high-mobility channel cases.
For this reason, we first generate the vector N_r which consists of a set of m values for N_r, each representing a different CQI report exchange rate. The first N_r value in the vector corresponds to a value noted $N_r^{(1)} = n$, which is the maximum number of CQI samples that may be collected during the prediction period.
To generate the rest of the values, we use the geometric progression method with ratio q as follows:

$$N_r^{(i+1)} = qN_r^{(i)} \tag{5.2}$$

In order to obtain decreasing N_r values, we assume that $0 < q < 1$ and select $q = 1/2$, which allows to observe the impact of decreasing N_r on E. Next, after creating the vector $N_r = (N_r^{(1)}, N_r^{(2)}, \ldots, N_r^{(m)})$, We can deduce the corresponding error of each N_r value and consequently deduce the error vector $E = (E_1, E_2, \ldots, E_n)$.
We proceed as follows: At the end of monitoring period τ, we collect vector CQI_τ, which we translate to the corresponding sequence of d_{pRB} values noted as $P_\tau = [p_1, p_2, \ldots, p_n]$. Then, out of this sequence of actual d_{pRB} values, for each $N_r^{(i)}$ we generate a sequence $\hat{P}_\tau^{(i)} = [\hat{p}_1^{(i)}, \hat{p}_2^{(i)}, \ldots, \hat{p}_n^{(i)}]$ of *estimated* ones by keeping only every $i-$th value from P_τ, and replacing the rest with their preceding real d_{pRB} value. For example, for $i = 2$ we have that $N_r^{(2)} = \frac{N_r^{(1)}}{2}$, thus $\hat{P}_\tau^{(2)}$ will be composed by keeping every second actual value from P_τ and setting every other value to its precedent, i.e., $\hat{P}_\tau^{(i)} = [p_1, p_1, p_3, p_3, p_5, p_5, \ldots]$. Finally, the error E_i is given by:

$$E_i = \frac{1}{n} \sum_{j=1}^{n} |p_j - \hat{p}_j^{(i)}|. \tag{5.3}$$

The idea behind the creation of N_r and E is to normalize them by their maximum and calculating the difference between their normalized values $\Delta_i = \mid \tilde{N}_r^{(i)} - \tilde{E}_i \mid$. Then, we select the N_r that corresponds to the minimum difference Δ_{min} between E and N_r, allowing to estimate the optimal N_r (N_r^{opt}) and, consequently, the optimal error. The steps of this method are illustrated in Figure 5.2.

5.3.2 Long Short-Term Memory method

Our second approach to reduce the reporting frequency is based on using the LSTM method to forecast d_{pRB} values for period $\tau + 1$, i.e., *without* actually collecting any

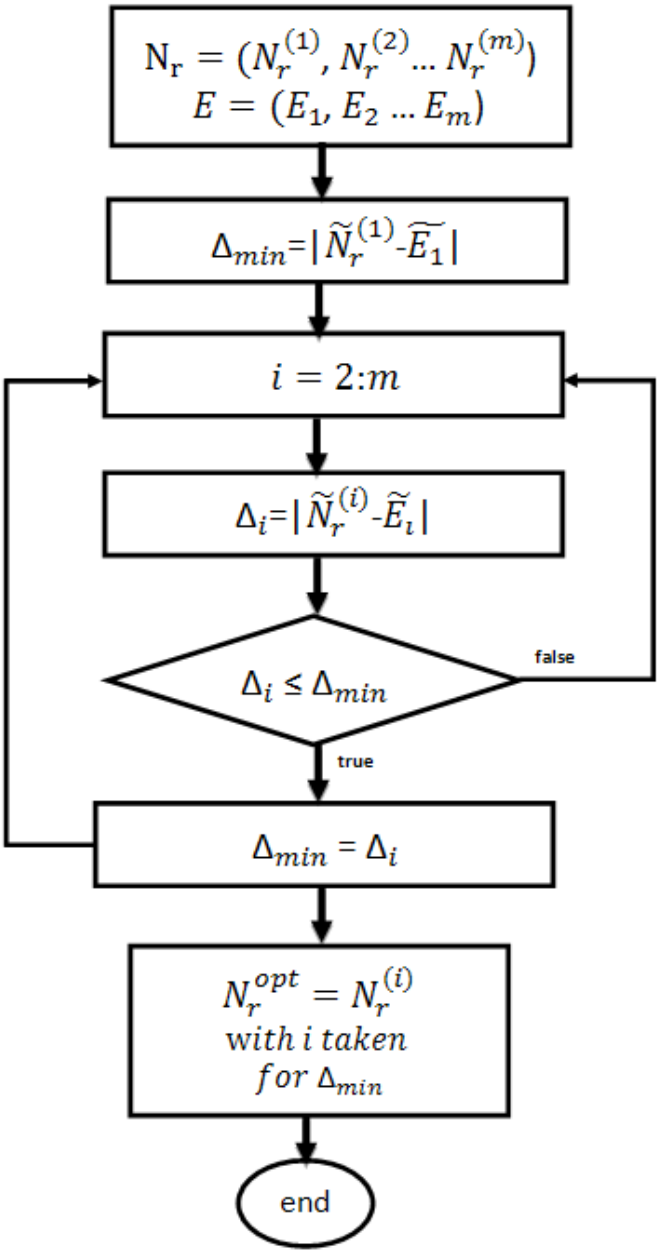


Figure 5.2: Steps to estimate the optimal number of CQI reports N_r^{opt}.

CQI values during $\tau + 1$.
LSTM is a deep learning method (a type of Recurrent Neural Network), which can learn the long term dependencies between time steps in time series, and sequence data. Its purpose consists in predicting the values of the future time steps of a sequence [99]. Therefore, in our case, the data sequence is constituted by the values of CQI or d_{pRB}, where we use LSTM to predict CQI or d_{pRB} values for period $\tau+1$ based CQI on d_{pRB} values collected during period τ.

5.3.3 Comparison between methods

After a monitoring period τ, where a sequence of CQI reports is collected, and the channel status is characterized, one of the two alternative strategies that we propose to reduce reporting frequency may be applied. The Optimal Difference method consists of estimating the appropriate number of CQI reports (N_r) to be collected during the next prediction period $\tau + 1$, which we consider here to have a fixed duration equal to that of τ, while the LSTM method consists in predicting a sequence of d_{pRB} values for $\tau + 1$ whose duration may vary according to the operator preferences, based only on the d_{pRB} values corresponding to the CQI reports collected during τ. Given the qualitative difference between the two methods, here we introduce appropriate metrics to compare them. First, we focus on the performance of each method in terms of CQI collection-related traffic savings. For this, we introduce two metrics representing the (normalized) CQI reporting rate gain for both methods following equations (5.4) and (5.5).

$$G_{OptDif} = \frac{CQIrate_{default} - CQIrate_{Opt}}{CQIrate_{default}} \tag{5.4}$$

$$G_{LSTM} = \frac{CQIrate_{Pred}}{CQIrate_{default}} \tag{5.5}$$

- $CQIrate_{default}$ refers to the number of CQI reports during period $\tau + 1$ with duration T without any optimizations (representing the maximum N_r).
- $CQIrate_{Opt}$ refers to the optimized number of CQI reports N_r in the same period.
- $CQIrate_{Pred}$ refers to the number of CQI values predicted during the LSTM forecasting period.

Second, we characterize the d_{pRB} estimation error for each strategy. In particular, for each case, the estimation error E during period $\tau + 1$ is defined as the mean (absolute) difference between a d_{pRB} value estimated and the actual one (if the corresponding CQI value were actually retrieved). We further normalize this error as follows:

$$E_{Norm} = \frac{E}{d_{pRBmax} - d_{pRBmin}}, \tag{5.6}$$

where $(d_{pRBmax} - d_{pRBmin})$ represents the difference between the maximum and minimum d_{pRB}, in order to express the maximum error that a method can detect in its prediction.
The purpose of these metrics is to capture the trade-off between gains in terms of CQI monitoring traffic reduction and d_{pRB} estimation accuracy, and help to identify via our quantitative evaluation the conditions under which each method is more appropriate.

5.4 Performance evaluation

This section focuses on evaluating the performance of the different methods and techniques proposed in this chapter. First, we evaluate the performance of the proposed algorithms for reducing the frequency of CQI reporting presented in Section 5.3, while keeping the d_{pRB} estimation error low. Then, we assess the efficiency of the data-driven RAN slicing algorithm introduced Chapter 3 (refer to Section 5.2), followed by a study on the impact of our mechanisms for reducing the reporting frequency on meeting RAN slice performance requirements.

5.4.1 Model validation channel mobility/stability prediction with real data

In Chapter 4, we evaluated the performance of the channel stability predictive system using two ML algorithms (NN and SVM). We generated CQI values for different channel mobility states using the `ns-3` simulator, where we simulated an LTE cell, considering UEs moving with different constant velocities, in order to create a data set with realistic CQI values corresponding to different degrees of user mobility. Therefore, we extracted a feature vector for each CQI vector as described in Chapter 4, which we labeled either as static or mobile, depending on the level of channel mobility. In this section, we validate our channel stability prediction mechanisms on real traces from an operating mobile network testbed. To this end, we used a data set publicly available from CRAWDAD [100], which contains statistics and monitoring data of 4G/5G MAC, RRC, and PDCP layers.

The considered data are raw and recorded for one eNB and a single mobile UE in five different mobility scenarios by following different motion and distance patterns relative to the eNB. All raw data have been recorded without including Tx power amplification on the RF front end (0 dBm transmit power), which implies an approximately 10 m maximum range of coverage.
From these data, we have extracted the CQI measurements for each one of the following mobility scenarios:

- Moving Away (MA): the UE moves away from the eNB to a maximum distance of 10 m.
- Moving Closer-Far-Closer (MC): the UE moves back and forth relative to the eNB, from a 0 distance up to approximately 10 m.
- Stable Long Distance (SLD): the UE stands still in a long-distance (approximately 5-10 m) away from the eNB.
- Stable Mid Distance (SMD): the UE stands still in a mid-distance (approximately 1-5 m) away from the eNB.
- Stable Short Distance (SSD): the UE stands still in a short distance (approximately 0-1 m) away from the eNB.

Then, we used CQI data that correspond to actual UE mobility scenarios to validate the channel stability results deduced previously using the `ns-3` dataset. Table 5.1 shows the confusion matrix results in a simplified way,[1] which indicates for each mobility scenario (MA, MC, SLD, SMD, and SSD), the results of the classification, i.e., mobile or static, obtained using NN and SVM.

Table 5.1: Mobility and stability results of the considered mobility scenarios using NN and SVM.

	NN	**SVM**
MA	*Static*	*Mobile*
MC	*Mobile*	*Mobile*
SLD	*Mobile*	*Mobile*
SMD	*Mobile*	*Mobile*
SSD	*Static*	*Static*

The obtained results indicate that both SVM and NN have correctly classified the channel of the MC scenarios as mobile. We argue this by the fact that the UE, in this case, is moving closer and away from the eNB, so the CQI values change with it. Also, for the SSD scenario, the channel was well classified as static since the UE remained static and close to the eNB.
For the SLD and SMD scenarios, both classification algorithms classified the channel as mobile, although the UE remained static. We explain this by the fact that for SLD and SMD scenarios, the distance between UE and eNB is long and medium, respectively, which strongly affects the channel by the undesirable effects between

[1] A confusion matrix indicates the number of true and false classifications across the whole validation dataset. As we have only 5 mobility scenarios, we had directly given the classification results obtained at the output of the ML algorithms used for each scenario.

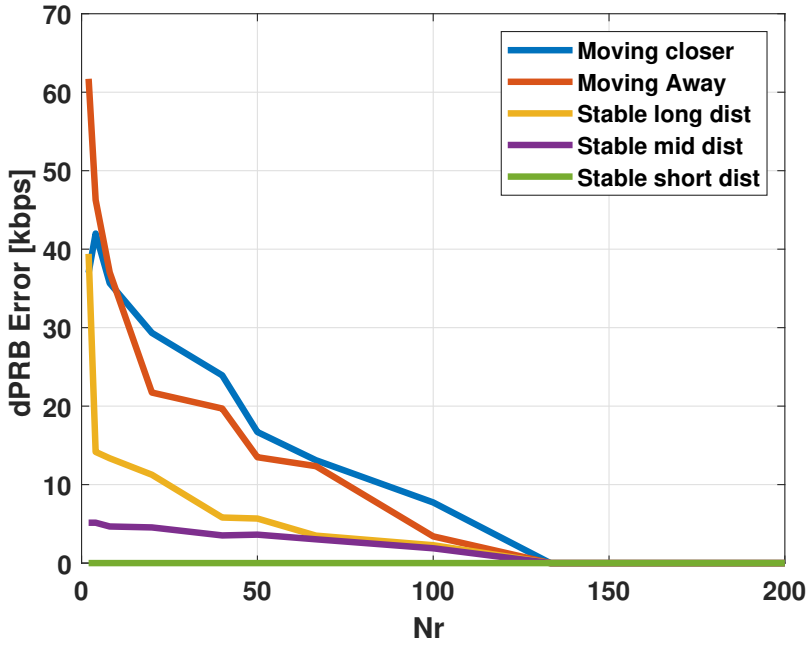


Figure 5.3: d_{pRB} error vs. N_r.

them, leading to significant variations in the channel quality. Regarding the MA scenario, the SVM correctly classified the channel as mobile, but the NN considered it as static. In this case, we cannot conclude that the NN does not work well since it is the only error in the prediction, and we have a very limited number of scenarios to generalize.

A. Optimal Difference method results

As elaborated in Section 5.3.1, the Optimal Difference method is a technique that calculates the estimation error of d_{pRB} for different CQI reporting frequencies, i.e., different numbers N_r of messages exchanged and, correspondingly, different inter-report times f_{rep}, over time intervals of length T. Then, it selects the N_R value that minimizes the difference between the (normalized) values of N_r and the pRB estimation error. In the time interval that follows, the optimal CQI reporting frequency, calculated as described above, is applied. An increase in N_r is equivalent to a decrease in f_{rep} according to Eq. (5.1).
Figure 5.3 shows the errors of the d_{pRB} values obtained for different N_r, in an interval of $T = 200ms$. The curves show that when N_r increases, the error decreases. They also demonstrate that the d_{pRB} estimation error for a UE in the MC scenario is the highest compared to the other scenarios for most CQI reporting frequencies. This error is higher than the MA scenario, followed by SLD, SMD, and SSD scenarios.
Once we have obtained the measures on the error according to N_r, we apply the optimal difference method presented in Section 5.3.1 to choose N_r^{opt}, i.e., the optimal

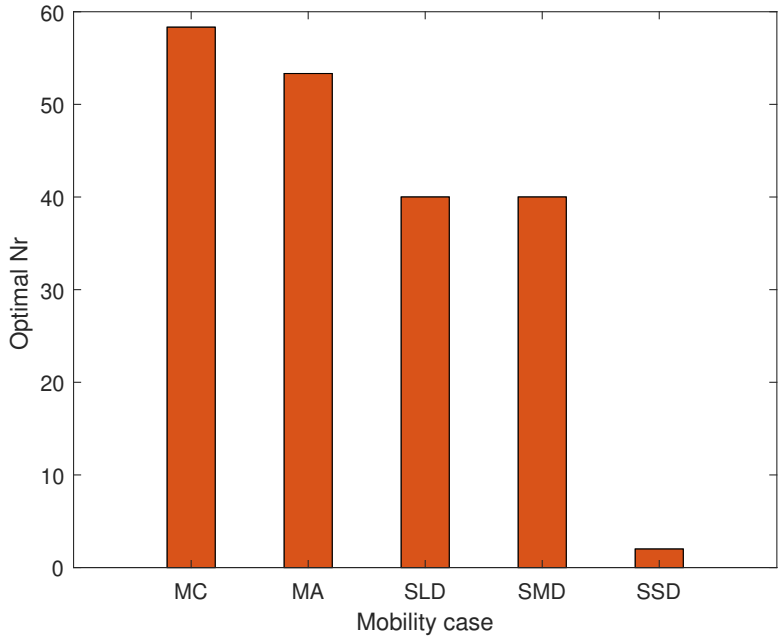


Figure 5.4: Optimal number of CQI reports (N_r^{opt}) over periods of $T = 200ms$ vs. mobility scenario.

value N_r. Figure 5.4 shows the N_r^{opt} value obtained for each scenario.
We notice that when the UE is mobile (scenarios MC and MA), our algorithm estimates that a significantly larger number/frequency of CQI reports is needed compared with the static scenarios (SLD, SMD, and SSD). For the latter, the error reduces as the distance decreases, and thus the optimal N_r decreases as well.

B. LSTM method results

Here, we present the results of the application of the forecasting method based on LSTM. After several tests, we selected in this LSTM network the Adam [101] solver for training, using 200 epochs. We argue for this choice by the fact that this configuration gave more precision at output. We trained the model on sequences of d_{pRB} values that correspond to collected CQI samples during 200 ms periods. Then, following a monitoring period τ of $T = 200$ ms where an input sequence is collected, we perform forecasting of the d_{pRB} values for the next period $\tau + 1$. The considered forecasting period duration are: 5, 10, 20, 50, 100 and 200 ms, where for each duration we calculate the corresponding d_{pRB} errors, as shown in Figure 5.5.
Similar to the previous results, we notice that the error of the MC and MA and sometimes the SLD scenarios are the highest, followed by the SMD and SSD. In addition, we observe that the longer the prediction period, the higher the error. We attribute this to the fact that the prediction time goes far beyond the actual values recorded to make the prediction.

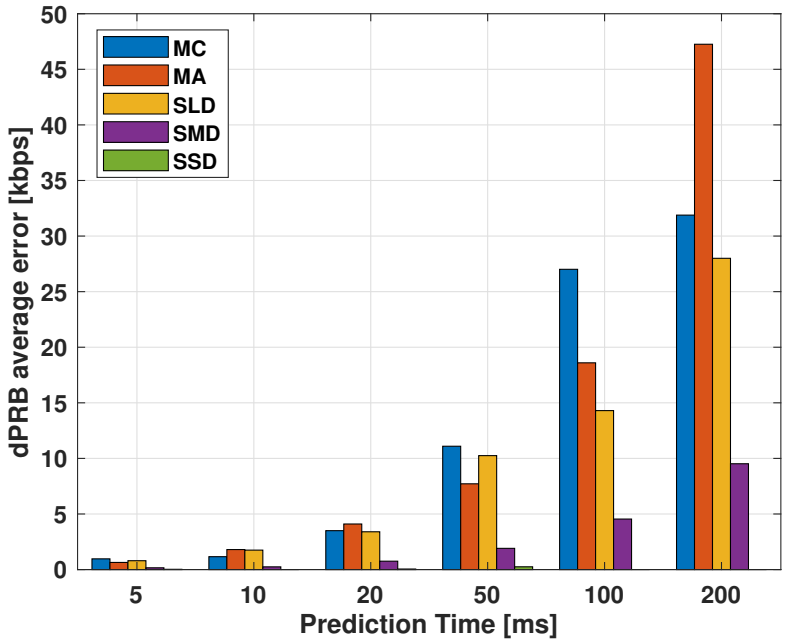


Figure 5.5: d_{pRB} Error vs. mobility state using LSTM.

C. Comparison between the Optimal Difference method and LSTM

We recall that the methodology and metrics used to compare between the two methods are detailed in Section 5.3.3. For the optimal difference method, a higher gain means fewer retrieved CQI reports during a prediction period of the same fixed duration as a monitoring period; the rest of the CQI values are generated as described in Section 5.3.1. For LSTM, this means a longer prediction period where we generated a sequence of d_{pRB} values using the learned model and with the sequence of values of the last monitoring period as input; the duration of this prediction period is left to the system operator.

In Figure 5.6, we draw the average normalized d_{pRB} error of the five considered scenarios against the normalized reporting gain. The latter is an expression of the number of CQI reports that are predicted (i.e., not actually retrieved) and thus represents the savings in terms of CQI monitoring traffic compared to the case where the default CQI collection takes place without any optimizations or prediction. As per the definitions of Section 5.3.3, for the LSTM method, When the number of predicted values equals the default number of CQI reports that would normally be collected during the period in question (in other words, when the prediction period has the maximum duration), the LSTM gain reaches 100%. On the other hand, for the optimal difference method, the gain depends on the optimal N_r value selected; the lower this value, the higher the gain.

We can observe that the optimal difference method performs better than LSTM in terms of error (up to 95% of gain). The error of the optimal difference method

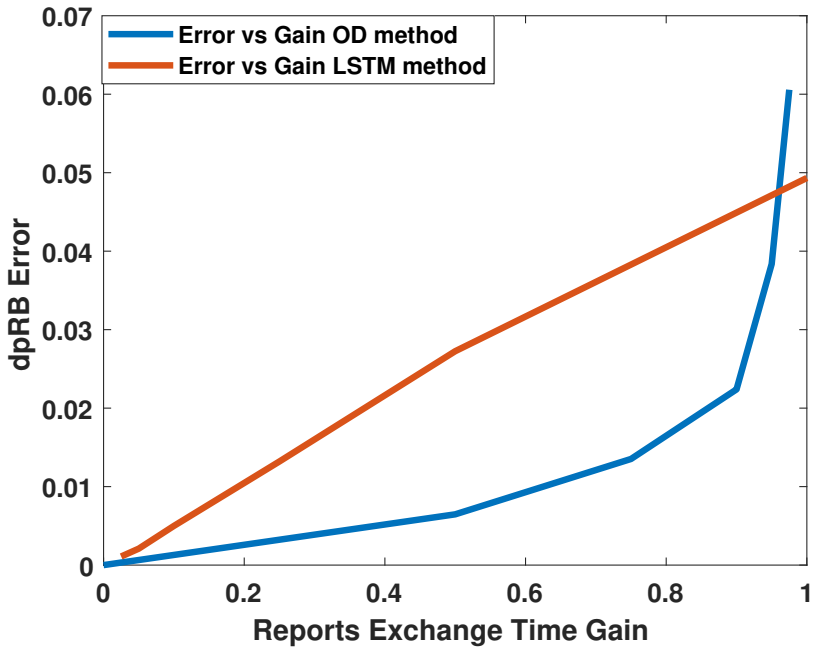


Figure 5.6: d_{pRB} normalized error vs. gain.

is smaller until reaching 95% of the gain; beyond this percentage, LSTM performs better. A d_{pRB} value predicted by the optimal difference method is always the same as the last actual value that corresponds to a real collected CQI sample. As the gain increases, the number of samples decreases, which drives the estimation error up. In such high-gain conditions, the values predicted by the LSTM model can better capture the actual variation of real d_{pRB} ones. Note that contrary to the LSTM method, which does not need to retrieve any actual CQI values during a forecasting period and can thus reach a gain of 100%, the optimal difference one, by design, always needs to retrieve *at least one* CQI value during its testing period (i.e., the number of CQI reports N_r cannot be zero) in order to consider it as the predicted CQI value for the rest of the interval when no CQI report will be collected.

5.4.2 RAN slicing with optimization

To evaluate the efficiency and the impact of the optimized RAN slicing solution on the slice requirements, we have integrated the CQI reporting frequency reduction algorithms to our mechanisms of resource sharing proposed in Chapter 3.
In this simulation, we consider the same parameters used in Chapter 3 (refer to Table 3.1) However, here each user has the real channel quality during the test period, using the real CQI measurements obtained from the CRAWDAD data set. The simulated results in this part are based on the calculation of the throughput of the eMBB slice, before and after the application of the CQI report reduction algorithms. Note that we have computed only the throughput of the eMBB slice,

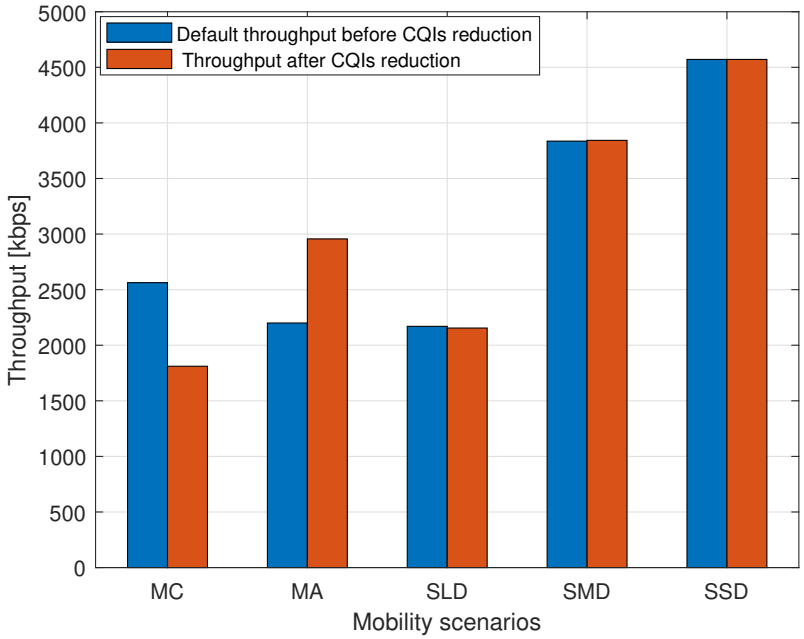


Figure 5.7: Throughput before and after applying the Optimal Difference method to reduce the CQI reporting frequency, for different user mobility scenarios.

as the throughput assigned to each slice depends directly on the number of pRB allocated to each slice. Thus we can show the optimization impact easily.
Figure 5.7 illustrates the default throughput assigned to the eMBB slice before applying any mechanisms to reduce the frequency of CQI reporting, and after applying the Optimal Difference method. The CQI report reduction algorithm was applied as follows: For each scenario, we calculated the optimal number of reports N_r^{opt} over a monitoring period and then we applied it over the period that follows. Finally, we calculated the throughput corresponding to this N_r^{opt}.

The results show that for the stable scenarios (SLD, SMD and SSD), the throughput assigned to each slice was not impacted by reducing the CQI reporting frequency considering N_r^{opt}. A small change is noticed in the two mobile cases MC and MA. In the MC case, less throughput was assigned after reduction, while in the MA case, more throughput was assigned. This error of assigning less throughput to MC and more to MA depends on the CQI values (high, medium or low) retrieved using N_r^{opt}.
The results of the CQI reporting rate reduction using LSTM are shown in Figure 5.8. Here, as there is no optimal number or interval reduction that would show the impact of LSTM, we calculate the average percentage of the throughput error by report to the real throughput. The latter is calculated for each scenario and uses all of the previously considered prediction intervals: 200, 100, 50, 20 and 5 ms (i.e., the average error between these time intervals). The obtained results show

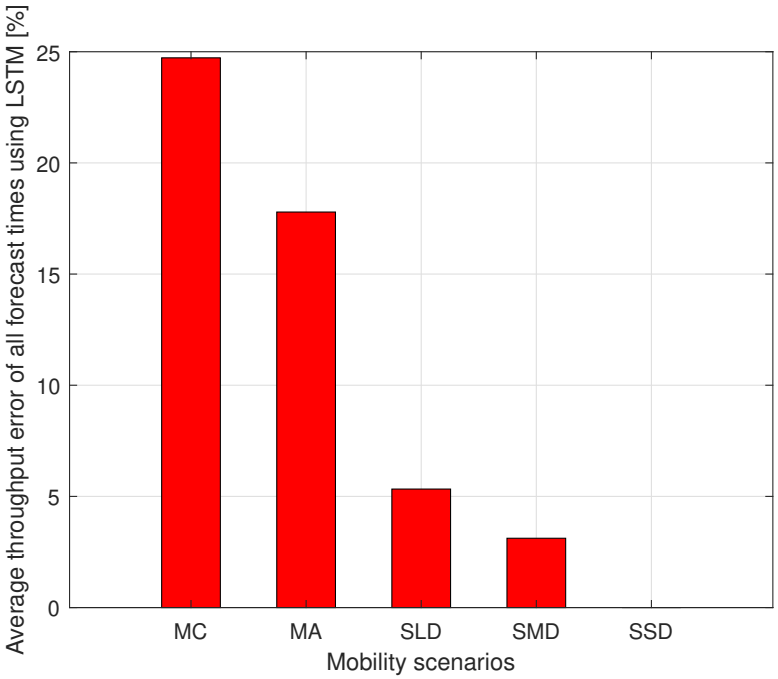


Figure 5.8: Average throughput error across all considered forecasting period durations using LSTM, for different user mobility scenarios.

that when user mobility increases, the LSTM cannot correctly predict CQI values. This has an impact on the slice requirements in terms of throughput since the error decreases when mobility decreases.

5.5 Conclusion

In this chapter, we proposed two predictive methods to mitigate the overheads associated with frequent CQI monitoring, i.e., *Optimal Difference* and *LSTM*-based forecasting. We also applied this decreased CQI frequency to adjust the radio resources needed by each running slice in the algorithms that drive the RAN resources among heterogeneous slices, proposed in our first contribution. The proposed algorithm runs at the SO level with very low complexity. Our objective was to reduce the frequency of CQI report exchanges between base stations and the SO, while minimizing the error of our estimates of the achievable throughput when the CQI reporting frequency is reduced.
Our simulation results demonstrate the positive impact of our CQI reporting optimizations by reducing the overhead while maintaining a precise prediction of RAN resources for the running network slices.

Chapter 6

On using reinforcement learning for network slice admission control in 5G: offline vs. online

Achieving a fair usage of network resources is of vital importance in Slice-ready 5G network. The dilemma of which network slice to accept or to reject is very challenging for the InfProv. On one hand, InfProv aims to maximize the network resources usage by accepting as many network slice as possible; on the other hand, the network resources are limited, and the network slices requirements regarding QoS need to be fulfilled.
In this chapter, we devise three admission control mechanisms based on Reinforcement Learning, namely Q-Learning, Deep Q-Learning, and Regret Matching, which allow deriving admission control decisions (policy) to be applied by InfProv to admit or reject network slice requests. We evaluate the three algorithms using computer simulation, showing results on each mechanism's performance in terms of maximizing the InfProv revenue and their ability to learn offline or online.

6.1 Motivations and state of the art

So far, in this thesis, we have addressed a critical challenge in RAN slicing, namely resource sharing between slices, and proposed methods to decrease the CQI signaling overhead between the SO and e/gNB. In this context, we have proposed two algorithms to drive the resources between slices, based on CQI reports. Our resource sharing algorithms were derived by the resources limit, where slices cannot have all their resource needs if resources are unavailable. Therefore, in this chapter, we present a policy that: (i) dealS with the resources' limitation, and (ii) introduce the concept of slice market.
It is worth noting that the network Slicing enables the emergence of new players in

the market: the Infrastructure, or slice, Provider (InfProv), which is the owner of the network infrastructure and may offer its resources as a service for a given cost, and the consumers, or tenants, requesting for network slice from InfProv to get a target service with specific needs [52] [102]. However, as each provider has limited resources [18] [77], it is challenging to have an optimal policy to decide which slice requests will be accepted (and/or rejected) by InfProv, and based on which criteria. Indeed, it is difficult to find the optimal policy that, on one hand, increases the revenue of the InfProv and allows an optimal usage of the infrastructure, and, on the other hand, guarantees the requirement of the admitted network slice in terms of QoS to avoid violating the SLA. Furthermore, the optimal admission control policy has to consider the long term income of InfProv by accepting the slices that maximizes the revenue while considering their traffic dynamics. For instance, it is difficult to decide between accepting network slices that pay a higher price for a long duration or accepting more network slices for a short duration but pay less money.

In this chapter, we propose novel slice admission control (SAC) algorithms to be run at the InfProv level aiming at deriving an optimal policy to decide if an arrived network slice request has to be accepted or rejected. The proposed algorithms are based on Reinforcement Learning and seek the optimal policy to increase the InfProv revenue while reducing the penalty to pay due to SLA violation. Three algorithms are introduced: Q-Learning (QL), Deep Q-Learning (DQL), and Regret Matching (RM). Besides deriving the optimal policy, we shed light on the proposed algorithms' ability to run offline or online, which is a crucial criterion. Indeed, offline solutions require a training phase before being used, which is sometimes costly, but they generally achieve the best results. Online solutions, on the other hand, are trained on the fly using only observable information of the controlled system.

In this context, different studies have addressed the problem of network slice admission control by exploring several techniques. In [18], the authors proposed an algorithm aims at maximizing the profit of InfProv, by admitting more slice requests than the overall capacity of the system, similarly to the concept of flight overbooking. They formulated the orchestration issue as a stochastic management problem that performs jointly resource allocation and admission control in all technological domains composing a mobile system. They then proposed solving the formulated problem using two algorithms: an optimal solution using Benders decomposition and a sub-optimal heuristic that accelerates the decision-making process. The authors of [103] proposed an admission control mechanism for network slicing that maximizes InfProv revenues while meeting services' latency requirements. They design a SAC policy using bid selection before studying the best strategy under different constraints (e.g., available resources, InfProv's strategy and requested traffic, etc.).

The authors of [102] proposed to maximize the revenues of InfProv when performing admission control by modeling the problem via Markov Decision Process

(MDP). They proposed two methods to solve the MDP: value iteration and Q-learning. This work aimed to maximize InfProv revenues by accepting the most expensive slices' requests first, which may lead to a lack of fairness between the network slices. Moreover, this work ignores the notion of priority between network slices, which, in contrast, will be considered in our contribution. Finally, it did not consider the needed physical resources in both UL and DL, which we will introduce in this work. In [104], the author proposed a new dynamic SAC model, using reinforcement learning. In this model, the InfProv generates revenues when accepting a slice request; and based on the requested slice priority, pays a penalty when rejecting it. The designed model aims at reducing the penalty fee by minimizing the rejection of expensive requests, hence maximizing InfProv revenues. This work also confirms the efficiency of using RL to maximize InfProv revenues.

However, in our work presented in this chapter, we consider more parameters to tackle the SAC issue, including the notion of hosting time needed by each slice and the requested physical resources by slice tenants at both UL and DL directions.

6.2 System model

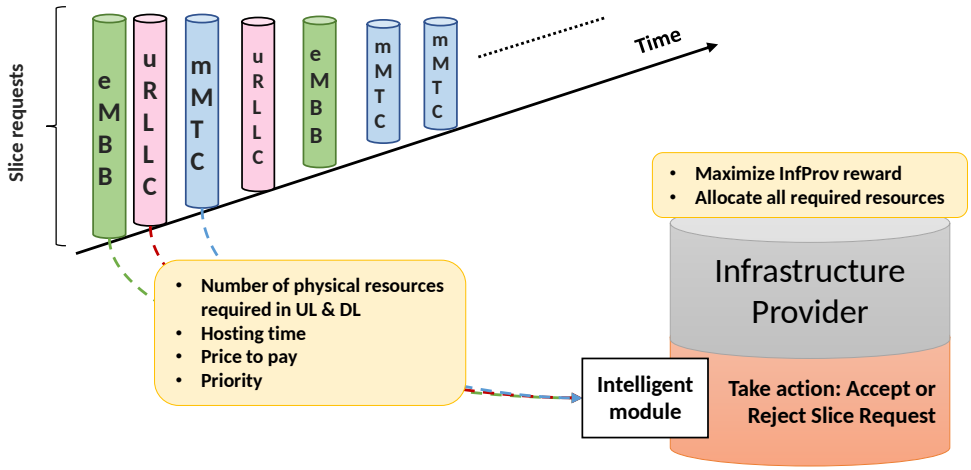


Figure 6.1: System model

In this section, we describe the considered system model, which is illustrated in Figure 6.1. It comprises the following actors:

Network players: In the proposed system model, we consider two main players: (i) the InfProv, which is the owner of the network infrastructure, and in charge of instantiating network slices for tenants by providing the required resources; (ii) the tenants that request the instantiation of the network slice from the InfProv to offer services for their clients.

Network slice model: In our model, each network slice is characterized by five main criteria:

1. The physical resources needed to satisfy the requirement of a network slice, in UL and DL, noted $Nres^{i}_{req}(UL)$ and $Nres^{i}_{req}(DL)$, respectively. Where i is the slice type, which can be either eMBB, uRLLC, or mMTC. We define the needed resources by each slice type as follows:

   - $Nres^{eMBB}_{req}(DL) >> Nres^{eMBB}_{req}(UL)$
   - $Nres^{uRLLC}_{req}(DL) >> Nres^{uRLLC}_{req}(UL)$ or $Nres^{uRLLC}_{req}(DL) == Nres^{uRLLC}_{req}(UL)$ or $Nres^{uRLLC}_{req}(DL) << Nres^{uRLLC}_{req}(UL)$
   - $Nres^{mMTC}_{req}(DL) << Nres^{mMTC}_{req}(UL)$

   We justify these assumptions as in eMBB DL traffic typically dominates (e.g. high definition video streaming), while in mMTC UL traffic is dominant (e.g. IoT traffic). The case of uRLLC is different, as all the types of traffic may exist and depend on the service.

2. The hosting time of each network slice type *Htime*: Each slice, if admitted, will use the InfProv's resources for a given duration.

3. *Priority* of the slice: It depends on the application running on the corresponding slice.

4. The price P^{i}_{req} that a slice tenant pays to InfProv for the used resources. The tenant has to pay the resources per time unit for the H_{time} duration. Here, req refers to a slice tenant, and i refers to the slice type.

InfProv model: The InfProv entity is characterized by its capacity in terms of available resources at time t (C_t). It represents the total amount of available resources that may be allocated to a new network slice. It is worth noting that C_t is updated when a network slice is admitted or leaves. In other words, when a new network slice is accepted, the needed resources will be allocated and dedicated to it; when a network slice ends, its associated resources will be automatically released. Thus, at each time instant t, the available resources at InfProv is performed as follows: $C_t = C_{total}$ - $C_{allocated}$ + $C_{released}$. It should be noted that two formulas are used, one for UL and one for DL, as the resources are separated. C_{total} is the total number of resources available in the InfProv in UL or DL. $C_{allocated}$ and $C_{released}$ are respectively the number of allocated and released resources at time t in UL or DL.

The proposed SAC model is applied only for the RAN resources, composed by a UL and DL. We argue this assumption by the fact that RAN is considered as the bottleneck of the system, while other network slice's required resources (such as computing) are always available, and no reservation is needed.
In summary, the InfProv is characterized by its resources capacity C_{total}, while a network slice is identified by: N^i_{res}, H_{time}, $Priority$, and P^i_{req}.

6.3 System Analysis

As stated earlier, we seek an optimal admission control policy that aims at finding a trade-off between fulfilling the network slice resource request (UL and DL), while maximizing InfProv revenues. First, we propose to model the SAC using MDP [105]. Since exactly solving the MDP is very challenging due to the difficulties in modeling the traffic dynamics, we apply reinforcement learning to derive the optimal policy and to find the earlier-mentioned trade-off. For that, we will use different Reinforcement learning models, namely QL, DQL, and RM, to predict the optimal action to apply when a new demand of a network slice arrives at the system (i.e., accept or reject an arrival slice request).

6.3.1 Markov Decision Process model

A Markov Decision Process is composed by 4-tuples $M = (S; A; T; R)$, where S is the set of states, A is the set of actions, T is the transition probability from state s at time t to state s' at time $t + 1$ when taking an action a, and R is the reward obtained by performing the action a, which leads to move from the state s to s'.
For our system, we assume that a state $s = (n, m, l, b)$ is composed of the following information where:

- n is the number of accepted eMBB slices;
- m is the number of accepted uRLLC slices;
- l is the number of accepted mMTC slices;
- b is a value that can be equal to 1,2 or 3 to indicate the slice type, eMBB, uRLLC, or mMTC, respectively, of the last received request.

At receiving a new network slice request, InfProv, via an agent, observes the state of the system and takes an action a either to accept or reject the request. The action set is as follows:

$$a = \begin{cases} 1 & \text{if new arrival slice request is accepted} \\ 0 & \text{if new arrival slice request is rejected} \end{cases} \tag{6.1}$$

The different transitions of the system occur when a new network slice arrives and a decision is needed, and when a network slice leaves the system. If the system is in a state $s = (n, m, l, b)$ and a new slice arrives, a decision needs to be taken (i.e. accept or reject), leading that the system transits to one of the following states:

- $(n + 1, m, l, 1)$ if a slice of eMBB is accepted;
- $(n, m + 1, l, 2)$ if a slice of uRLLC is accepted;
- $(n, m, l + 1, 3)$ if a slice of mMTC is accepted;
- $(n, m, l, 1)$ if a slice of eMBB is rejected;
- $(n, m, l, 2)$ if a slice of uRLLC is rejected;
- $(n, m, l, 3)$ if a slice of mMTC is rejected.

If a network slice leaves, then the system moves to one of the following states, without taking any action:

- $(n - 1, m, l, 1)$ if a slice of eMBB has left;
- $(n, m - 1, l, 2)$ if a slice of uRLLC has left;
- $(n, m, l - 1, 3)$ if a slice of mMTC has left.

As mentioned above, each network slice is described by $<Nres^i_{req}(UL)$, $Nres^i_{req}(DL)$, H_{time}, *priority* and $P^i_{req}>$. We assume that the price P^i_{req} to pay by each slice tenant, by time unit, is proportional to the slice *priority*. Hence, we propose to model the estimated reward that InfProv expects to receive from each accepted network slice as follows:

$$R_{inf} = sign(C_t - Nres^i_{req}) \times P^i_{req} \times Htime^i_{req} \tag{6.2}$$

With:

$$sign(C_t - Nres^i_{req}) = \begin{cases} 1 & C_t \geq Nres^i_{req} \\ -1 & C_t < Nres^i_{req} \end{cases} \tag{6.3}$$

In equation 6.2, we multiply the P^i_{req} by H_{time}, since each accepted slice tenant pays a P^i_{req} according to the slice priority by a time unit. Hence, the total price that a tenant of an accepted slice will pay depends on his priority and the requested hosting time. Besides, we have added the sign in this equation to ensure that there are enough resources in InfProv to support the number of required slice resources.

It is worth noting that in this work, for each accepted network slice, the InfProv should be able to provide the needed resources for both UL and DL. Otherwise, the InfProv will pay a penalty, if it accepts a slice request without having enough resources to cover the slice resource requirements. To this end, C_t should be always

higher than $Nres^i_{req}$ in UL and DL. Therefore, the reward defined in 6.2 for both UL and DL will be calculated as follows:

$$R_{inf} = [sign(C_{InfDL} - Nres^i_{reqDL}) \wedge sign(C_{InfUL} - Nres^i_{reqUL})] \times P^i_{req} \times Htime^i_{req} \tag{6.4}$$

We also note that the R_{inf} is null if the slice request is rejected, as neither penalty nor reward can be applied. Having defined the MDP model, we need to find the optimal policy that maximizes the long term total reward for InfProv. The optimal policy corresponds to the action to take for each state s aiming at maximizing the long-term total reward. Since the MDP is hard to solve using techniques like Value iteration or Policy iteration as the traffic model is hard to model, we describe in the next section how to find this policy using Reinforcement Learning, through three models QL, DQN, and RM.

6.3.2 Admission control using QL

Q-learning is an offline reinforcement learning algorithm that generates an optimal policy to maximize the expected total reward for any finite MDP, i.e., the state and action spaces may be finite, which is our model's case. This policy is based on the Q-learning function, which is designed to seek the best action in each state to maximize the long-term total reward.

The QL method consists first of calculating, for each possible action in each state, a value named Q-value. Then the QL method stores these Q-values in a table, namely the Q-table. This step is called the exploration of the unknown environment. It should be noted that the Q-table is initiated to zero and updated with the new Q-values obtained after each episode.

The agent performs in a state s_t, one of the two actions: accept or reject a new slice request for the epoch t and it observes the state transitions s_{t+1}, and rewards r. Hence, it updates the Q-value using the weighted average of the previous and the new Q-value, as shown in the following equation:

$$Q_{new}(s_t, a_t) \leftarrow (1-\alpha)\, Q(s_t, a_t) + \alpha \left(r_{t+1} + \gamma \max_{a \in A} Q(s_{t+1}, a) \right) \tag{6.5}$$

with:

- $Q(s_t, a_t)$ is the old Q value;
- $Q_{new}(s_t, a_t)$ the new value obtained after updating the old one;
- α is the learning rate that controls how fast the new estimation adapts to the random changes imposed by the environment.
- γ is the discount factor that notifies the importance of future rewards;

- r_t is a reward received from action a_t;

- $maxQ(s_{t+1}; a)$ is the estimation of the optimal future action.

After several episodes, the Q-table converges and becomes the reference table for the agent (i.e., the entity that takes decisions) to select the best action based on the Q-value. However, one of the QL method's weaknesses is the convergence time, i.e. the time needed by the agent to explore all the states to learn the best action to take in the future. Indeed, it depends on the state space; if the latter is big, the time to converge is high, which may be problematic if QL is used without offline training.

Algorithm 3: Deep Q Learning algorithm

Ensure: $s \leftarrow s(t+1)$
Initialize: replay memory D to capacity N
Initialize: $Q_0(s, a)$ with random weights,
for *episode* $\leftarrow$ *1: M* **do**
  0: Initialize state s_t
  for $t \leftarrow$ *1: T* **do**
    - With probability ϵ select:
      - a random action a_t
      - a_t= max_a Q(s_t, a, θ) (otherwise)
    - Execute action a_t and observe reward r_t and state s_{t+1}
    - Store transition (s_t,a_t,r_t,s_{t+1}) in D
    - Set s_{t+1}= s_t
    - Sample random minibatch of transitions (s_t,a_t,r_t,s_{t+1}) from D
    - Set $Q_{targetj}$= $\begin{cases} \text{r}_j \text{ for terminal } s_{t+1} \\ \text{r}_j + \gamma * max_{a'} * Q(s_{t+1},\text{a'},\theta_{i-1} \text{ for non terminal } s_{t+1} \end{cases}$
    - Perform a gradient descent step: $MSE(\theta_i)$= $[Q_{target} - Q(s_t, a_j, \theta_i]^2$
  end
  =0
end

6.3.3 Admission control using DQL

Q learning is based on a Q-table to store the learned results for each state and action. Consequently, if the state space is big, the table size explodes, leading to an increase in the training time as the agent has to take more time to explore all the states.

To this end, DQL uses deep learning to represent the Q-values, where each state passes through several hidden layers of a neural network to get the Q-values. Then, DQL calculates: (i) the loss function that represents the MSE as shown in equation 6.6 of the predicted Q-value (Q_{pred}), and (ii) the target Q-value (Q_{target}) (see equation 6.7) that represents the maximum possible value for the next state.

$$MSE(\theta_i) = [Q_{target} - Q(s_t, a, \theta_i)]^2 \tag{6.6}$$

$$Q_{target} = E[r + \gamma max Q(s_t, a', \theta_{i-1})] \tag{6.7}$$

With θ is the weight.

Using the same θ weights in equation 6.6, the values Q_{target} and Q_{pred} move at the same time. For this purpose, DQL uses two neural networks, one for Q_{pred} and the other one for Q_{target}. Algorithm 1 presents the different steps of the DQL algorithm.

Note that we have considered the states and actions as defined in the MDP.

6.3.4 Admission control using Regret Matching

RM is an online learning algorithm similar to Reinforcement Learning. Its agent (player or user) looks for the right action based on the regrets of the previous actions. The main principle consists in minimizing the regrets of its decisions at each step of the system. To do so, the agent relies on past behavior of taken actions to guide its future decisions by favoring the actions that it regrets not to have chosen before.

The strategy of this method is to adjust the agent's policy by distributing probabilities on all actions proportionally to the regrets of not having played other actions. The regret is defined as follows: if a is the action chosen by the agent at time T, thus for any other action $a \neq a^*$, the regret of choosing the action a but not another action a^* up to time T is obtained as shown equation 6.8 [106].

$$Reg_T(a, a^*) = \frac{1}{T}\sum_{t=1}^{T} r_t(a) - \frac{1}{T}\sum_{t=1}^{T} r_t^i(a^*) \tag{6.8}$$

with: $r_t(a)$ is the reward obtained at time T, by choosing the action a. At each step, the agent chooses an action a_T between two actions (accept or reject) considered in this study (see equation 1). The probability P_{T+1} that the agent will choose action a in the next step defined by next time T+1, is defined as follows:

Algorithm 4: Regret matching algorithm

Result: $s \leftarrow s(t+1)$

Initialization: action a_1,

for $t \leftarrow$ *1:* T **do**

- Take action a_t
- Receive reward $r_t(a)$ and $r_t(a^*)$
- Update the regret of choosing action a_t and not a_t^* is: $\mathrm{Reg}_T(a, a^*) = \frac{1}{T}\sum_{t=1}^{T} r_t(a) - \frac{1}{T}\sum_{t=1}^{T} r_t^i(a^*)$
- The probability of choosing action a_t in the next step is: $\mathrm{P}_{T+1}(a) = \begin{cases} \frac{[Reg_T(a,a^*)]^+}{\sum_{a=0,1}[Reg_T(a,a^*)]^+} & \text{if a} \neq \mathrm{a}_T \\ 1 - \sum_{a' \neq a_T} P_{T+1}(a') & \text{if a}= a_T \end{cases}$
- Select action corresponding to maximum probability
- Update action a_{t+1}

=0

end

$$P_{T+1}(a) = \begin{cases} \frac{[Reg_T(a,a^*)]^+}{\sum_{a=0,1}[Reg_T(a,a^*)]^+} & \text{if a} \neq \mathrm{a}_T \\ 1 - \sum_{a' \neq a_T} P_{T+1}(a') & \text{if a}= a_T \end{cases} \tag{6.9}$$

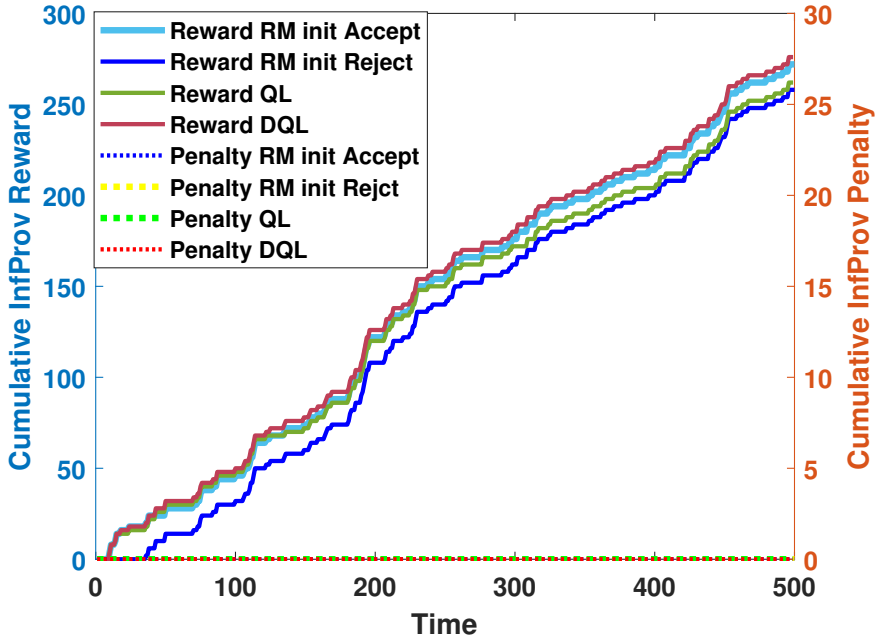
With: $Reg_T(a, a^*)]^+ = \max\,[Reg_T(a, a^*),0]$ presents the non negative part of the regret $Reg_T(a, a^*)$.

The RM algorithm is illustrated in Algorithm 2.

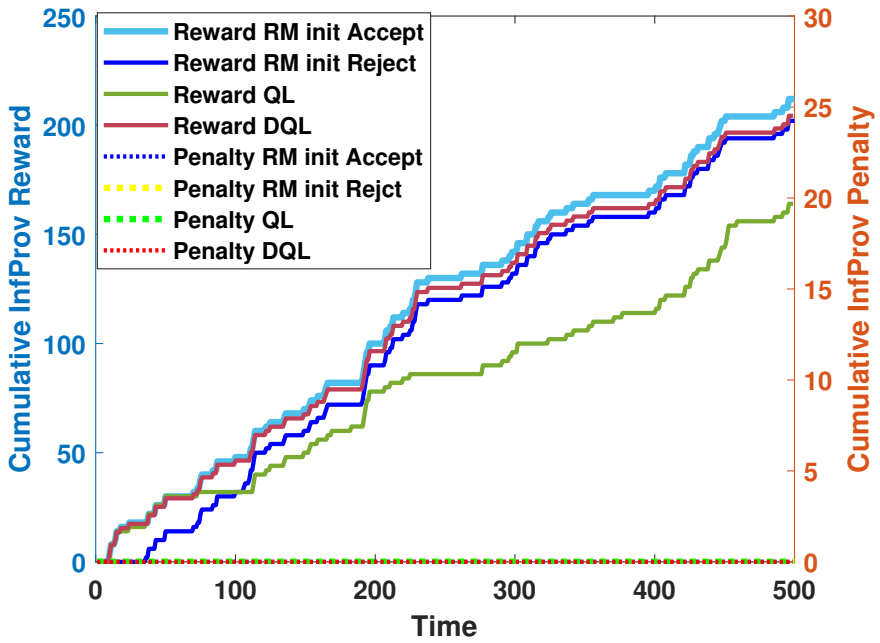
It should be noted that the RM is a fully online solution; the policy should be initiated before that the algorithm starts adapting itself. Therefore, we consider RM with two initial policies: accept and reject. The first one starts by accepting the network slice requests, while the second one starts by rejecting the requests.

6.4 Performance evaluation

In this section, we present the simulation results of the slice admission control problem by comparing the three methods' performances. It is worth noting that the RM considered here is initialized once by accepting the first received requests (noted as RM with accept policy), and once by rejecting the first received requests (noted as RM with reject policy).



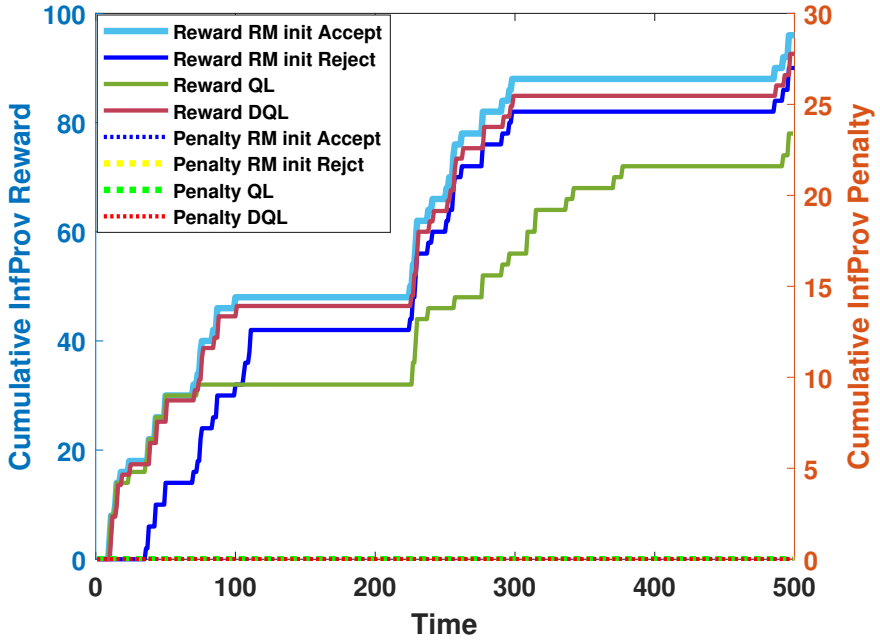
(a) H_{time}= 5 tu



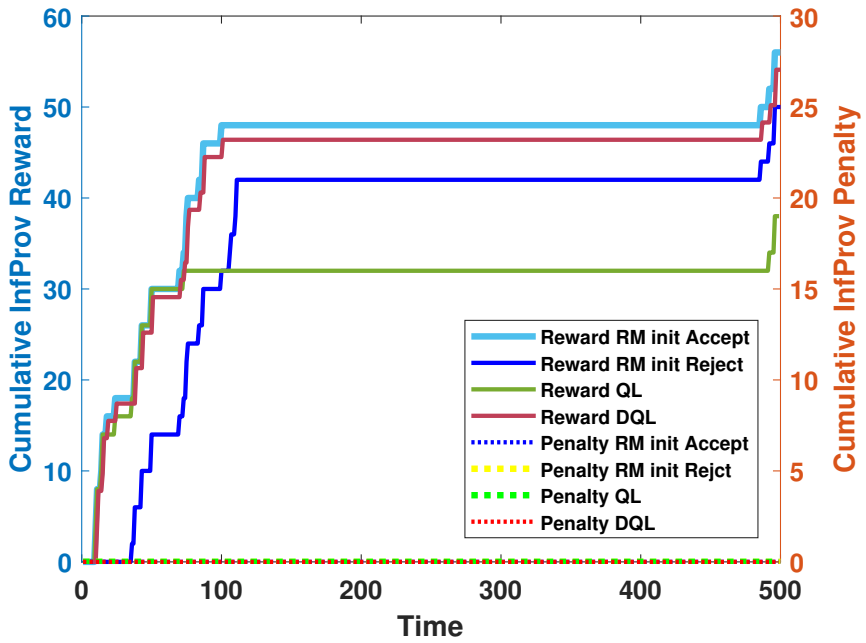
(b) H_{time}= 20 tu

Figure 6.2: $InfProv$ reward and penalty vs. $Time$ for slice arrival request rate= 2, and H_{time}= 5 & 20 tu

We assume that InfProv receives requests to create slices following a Poisson process with two different arrival rates as follows: (i) rate=2 per time unit (tu) corresponding to a low arrival rate (i.e. the slice requests arrive rarely), and (ii)



(a) H_{time}= 50 tu



(b) H_{time}= 100 tu

Figure 6.3: $InfProv$ reward and penalty vs. $Time$ for slice arrival request rate= 2, and H_{time}= 50 & 100 tu

rate=10 per tu corresponding to a frequent slice request arrival. Besides, we assume that each slice request stays hosted in InfProv for a H_{time} period. To show the impact of H_{time}, we use four values as follows: (i) short period where H_{time}= 5 *tu*,

Table 6.1: Number of resources: (i) available in InfProv, (ii) requested by each slice in UL and DL

[UL, DL] InfProv	[100, 100]
[uRLLC, mMTC, mMBB] UL slices	[5,9,5]
[uRLLC, mMTC, mMBB] DL slices	[5,5,10]

(ii) medium period where $H_{time} = 20$ *tu*, (iii) large period where $H_{time} = 50$ *tu*, and (vi) very large period where $H_{time} = 100$ *tu*. We consider that the number of resources requested by each slice in UL and DL, and the number of resources available in the InfProv are different, and their values are presented in Table 6.1.

Regarding slices' priority, we assume that uRLLC slices have the highest priority, while eMBB and mMTC slices have the same priority. The price of running uRLLC slice type is four times higher than the price to pay for running the mMTC and eMBB slice types. The latter have the same price. In other words, we use the price to pay as a way to enforce priority among the slice types.
Figures 6.2 and 6.3 illustrate the cumulative reward as well as the cumulative penalty obtained using the proposed algorithms (RM initialized with accept policy, RM initialized with reject policy, QL, and DQL), when the arrival rate is 2 per *tu*, and for four values of the holding time (H_{time}). The same metrics are shown in Figures 6.4 and 6.5, but for an arrival rate of 10 per *tu*. We recall that cumulative reward is obtained by the InfProv when accepting slices, and the penalty is incurred when a slice is accepted but InfProv has not sufficient resources either in DL or UL, or in both directions to satisfy its requirements. For the cumulative reward, we notice that, for both arrival rates and in the four proposed solutions, when the H_{time} increases, the cumulative reward decreases. We argue this by the fact that the resources are not released quickly when H_{time} is high; hence the InfProv rejects new requests during this H_{time} period.
Besides, penalties (accepting a slice without having a resource) occur when the InfProv resources are saturated, and the SAC keeps accepting arrival slices. We remark that penalties are higher when both the arrival rate and the holding time are high, which is evident as the resources are quickly saturated since accepted slices stay longer in the system. Further, we note that most of the algorithms require some time to detect that the resources are saturated and keep accepting requests until the reward starts to be negative. Consequently, they change the policy to reject. The time to detect that the resources are saturated is a criterion to understand which algorithm performs well, and hence learned the system's behavior. In this case, we remark that RM obtains the best performances with accept policy, followed by DQL. QL achieves the worst performance. We argue this by the fact that RM, thanks to its regret formula, detects quickly that the reward starts to be negative,

and adapts the policy accordingly. Further, DQL with the neural network can learn and predict better when to change the policy, compared to QL, where the Q-tables cannot predict when to update.
On the other hand, when the arrival rate and the holding time are low, the probability of having penalties is very small or even null as there are always available resources to accept new slices as shown in Figures 6.2 and 6.3.
The results of Figures 6.2, 6.3 and Figures 6.4 and 6.5 also show that in terms of rewards, the QL algorithm is the worst algorithm of all tested algorithms regardless of the arrival rate of each type of slices, by achieving the lowest cumulative reward. This means that it does not learn well when the policy should change from accepting to rejecting, or the contrary. Indeed, QL derived policies that favor rejecting slice requests.

Figure 6.6 presents the percentage of rejected requests according to H_{time}, when using the proposed algorithms: RM with accept policy, QL and DQL, and a request arrivals rate= 10, corresponding to a frequent slice request arrival.

We notice that increasing H_{time} leads to an increase in the percentage of rejection for all the algorithms. This is obvious as high values of H_{time} mean that admitted slices will stay longer in the network, and hence low resources are available to accept new slices (i.e. increase the reject rate). Moreover, we notice that QL rejects more requests than RM and DQL for all the H_{time}, which confirms the low cumulated penalty and reward of QL shown in the two precedent figures.

In Figures 6.7 and 6.8 we present the percentage of accepted slices according to their type and for different H_{time}, when using RM algorithms with accept policy, QL and DQL. Here our objective is to verify whether the proposed algorithms satisfy the slice priority condition, i.e. the ability to give priority to uRLLC slices compared to the other slice types. We can see that all algorithms favor uRLLC slice over the other slice types. Further, we see that DQL and RM show the highest percentage of accepted uRLLC slices compared to QL. In other words, these results validate our reward function and the fact of using the price to pay as a way to enforce priority among the three types of slices.

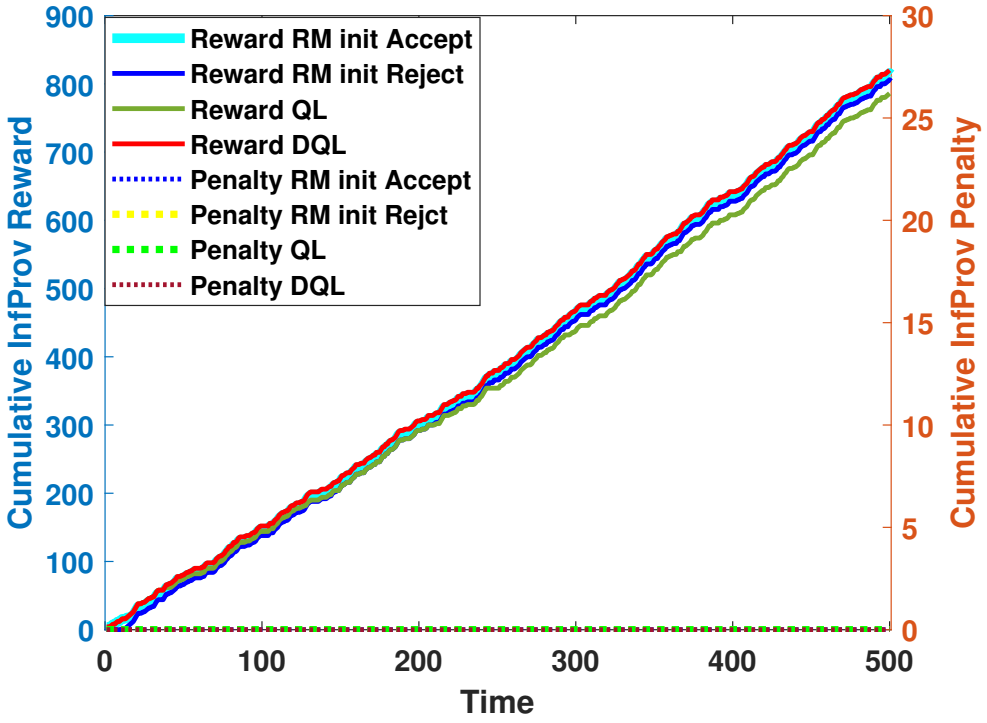
One of the biggest challenges when using Reinforcement Learning to solve SAC's problem is whether online or offline learning is better? And what is the time of convergence? So far, we have seen that RM, a fully online algorithm, works well for SAC's problem, while DQL and QL need to be trained before being used. Regarding DQL we wanted to understand if an online version could make sense to address the SAC problem efficiently. To this aim, we draw in Figure 6.9 a comparison between the offline DQL (when we first perform the training phase and then the tests) and the online DQL in terms of average reward. The online training phase of DQL varies between 0 and 1000 episodes. Each episode represents one training epoch during which the system receives 100 arrival slice requests. The maximum number of episodes depends on the convergence of the online learning model. We

have increased the number of episodes and calculated the corresponding average reward. We have stopped when the learning model starts to give a constant reward (between 500 and 1000). Note that the average reward is an average value of the reward obtained for the four considered H_{time}. We clearly observe that the online DQL needs time i.e., more episodes to converge (improve the learning) and start achieving the same performance as offline DQL (around 400 episodes). This means that during this period, i.e., before converging, the DQL performances are awful and can seriously affect the business of InfProv.

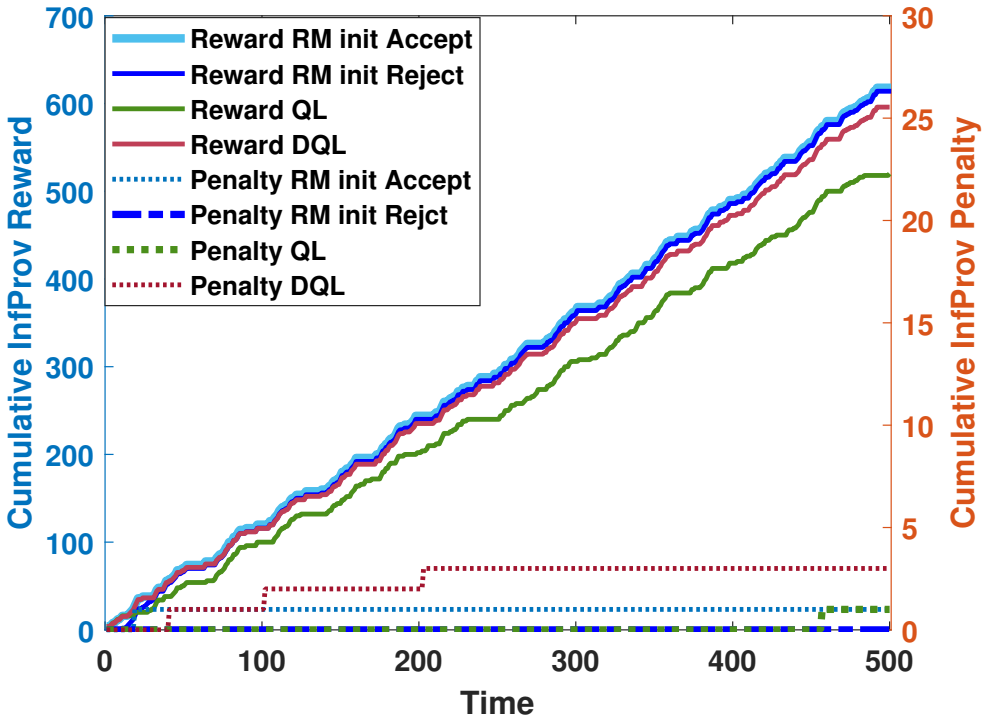
All the results confirm that one of SAC's best policy is to accept whenever the resources are available to maximize the InfProv profile. In this context, RM with accept policy achieves the best performance by reducing the penalty and increasing the reward. DQL and QL could be a good candidate, but there is a need to well tune the learning steps in order to anticipate when the policy has to change. Indeed, RM uses a simple and efficient formula to understand the need to change policy, while DQL and QL need to learn this. However, in a more complex system, where a high number of actions are available, things may change as RM can hardly, by using a simple formula, capture the behavior of the system. In contrast, DQL can be a powerful solution. But, in the case of a SAC with only two available actions, RM with accept policy is the best alternative for InfProv.

6.5 Conclusion

In this chapter, we have studied the challenge of network slice admission control for future 5G networks. We have proposed algorithms based on Reinforcement Learning (QL, DQL), and an online algorithm, which is the Regret Matching, to solve this problem. We first modeled the problem as a MDP and described how the proposed algorithms could derive a policy to be used by the agent to maximize the InfProv revenue while avoiding violating network slice requirements. We have modeled the reward as a function of the network slice price to pay and the penalty to pay if the slice requirement is not satisfied. Through extensive simulation, we showed that all algorithms could derive a good policy, but RM achieves the best performance, as the SAC problem is not complex, and the actions state is limited.

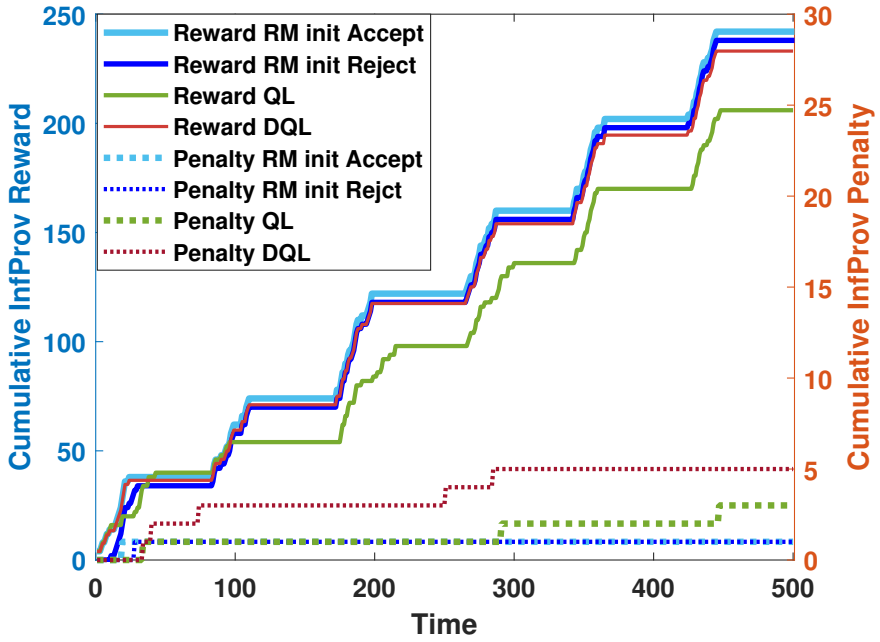


(a) H_{time}= 5 tu

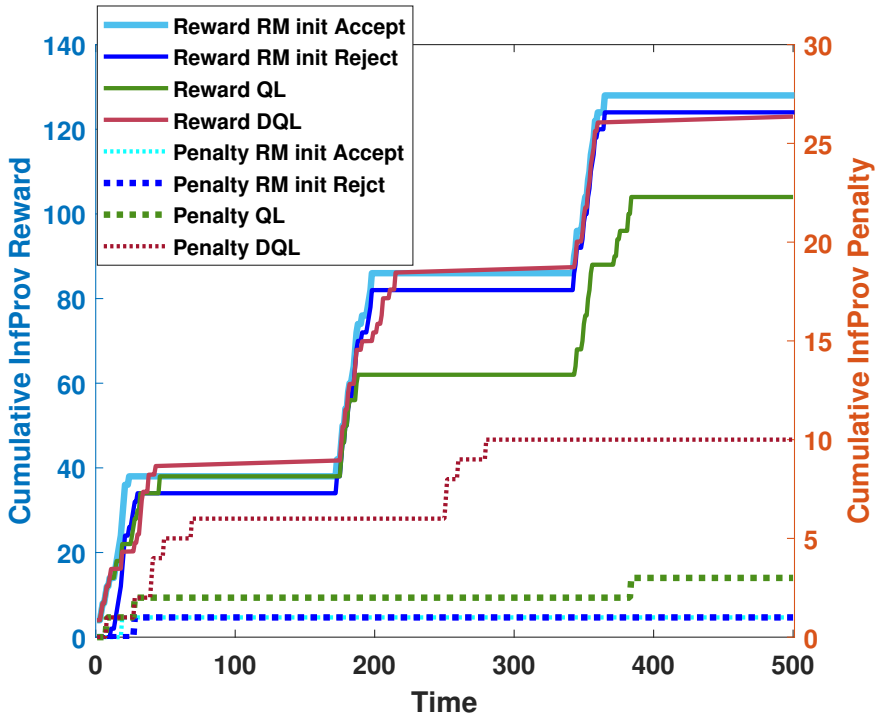


(b) H_{time}= 20 tu

Figure 6.4: $InfProv$ reward and penalty vs. $Time$ for slice arrival request rate=10, and H_{time}= 5 & 20 tu



(a) H_{time}= 50 tu



(b) H_{time}= 100 tu

Figure 6.5: $InfProv$ reward and penalty vs. $Time$ for slice arrival request rate=10, and H_{time}= 50 & 100 tu

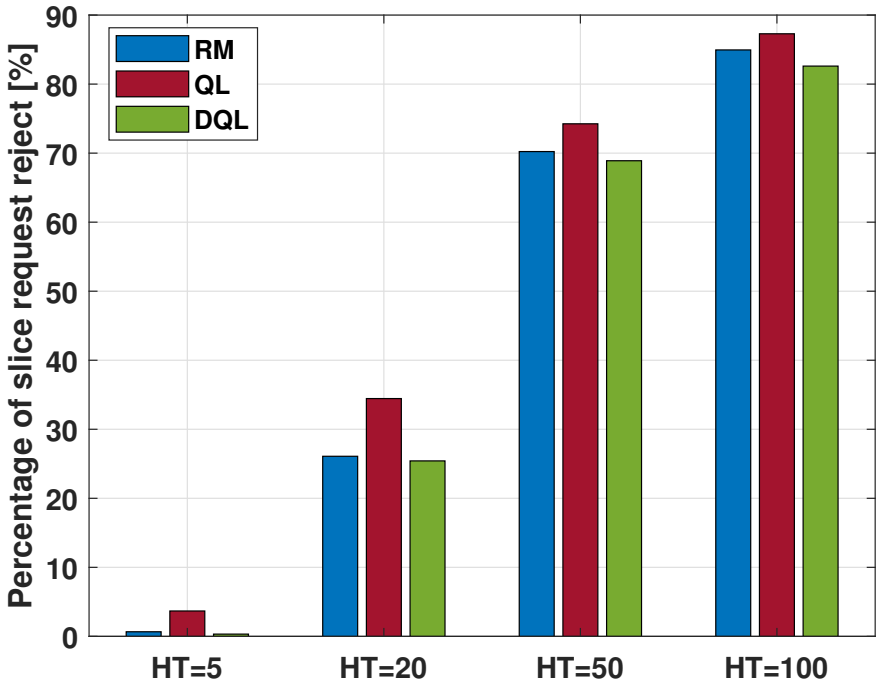
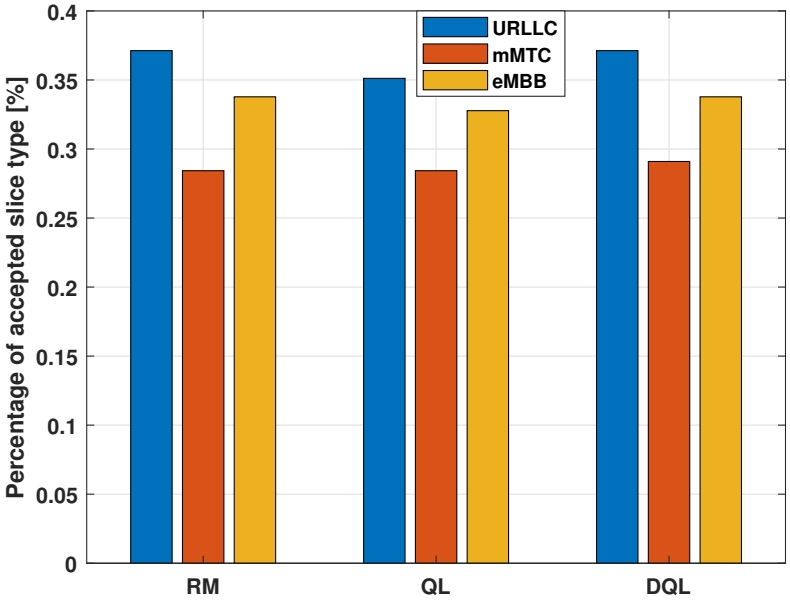
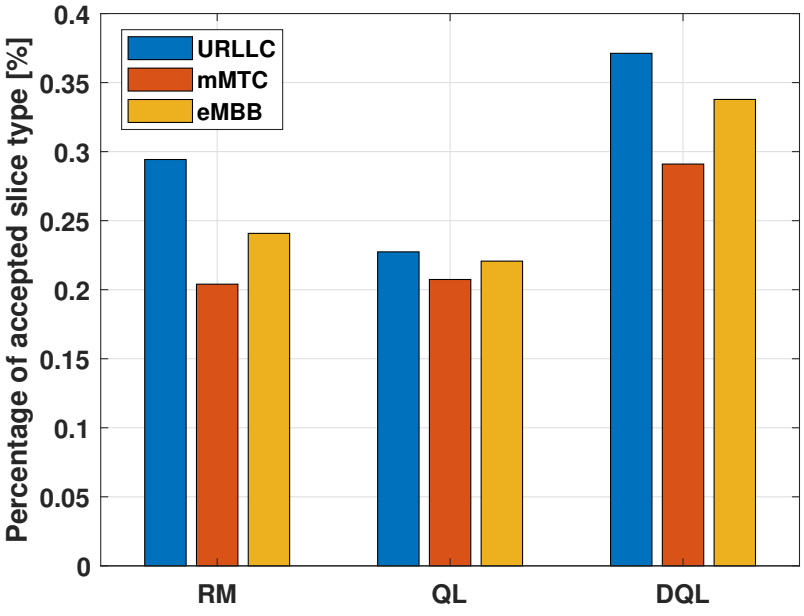


Figure 6.6: Percentage of slice request reject vs H_{time} for slice arrival request rate= 10

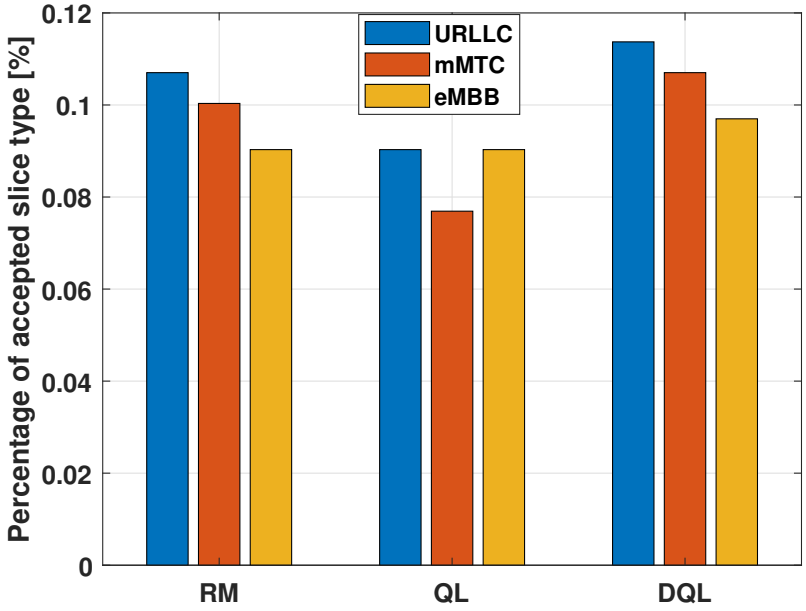


(a) H_{time}= 5 tu

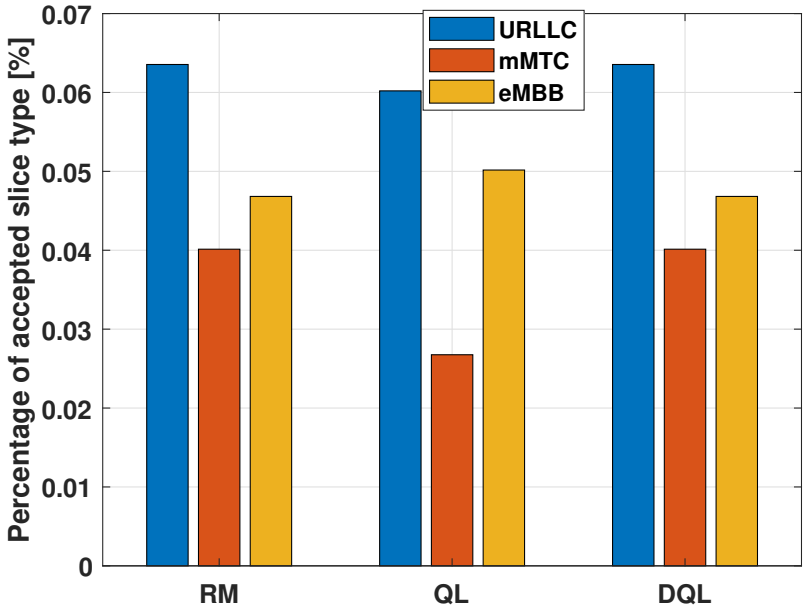


(b) H_{time}= 20 tu

Figure 6.7: Percentage of accepted slice type for slice arrival request rate=10, and H_{time}= 5 & 20 tu



(a) H_{time}= 50 tu



(b) H_{time}= 100 tu

Figure 6.8: Percentage of accepted slice type for slice arrival request rate=10, and H_{time}= 50 & 100 tu

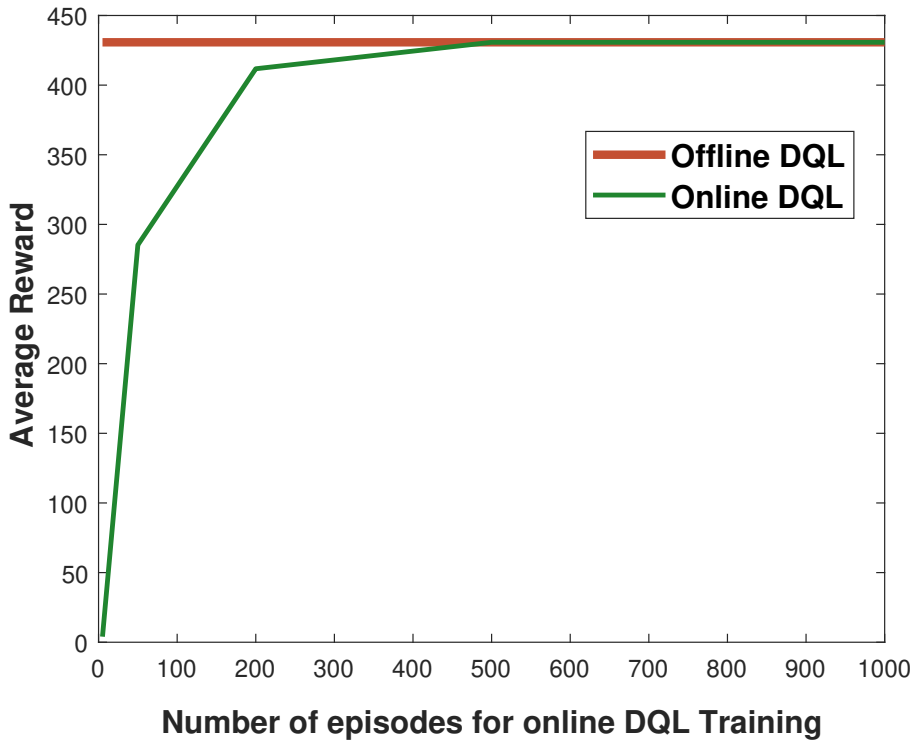


Figure 6.9: Offline and Online DQL Average Reward of H_{time}= (5, 20, 50, 100) for slice arrival request rate= 10

Chapter 7

Conclusions and Perspectives

This chapter summarizes and lists important conclusions. Besides, different perspectives left for future work are discussed in the second part of this chapter.

7.1 Conclusions

7.1.1 Dynamic slicing of RAN resources for heterogeneous coexisting 5G services

The first contribution of this thesis addressed the problem of resource sharing and isolating between the heterogeneous network slices in 5G.
In this contribution, we have proposed two algorithms that estimate the needed RAN resources for two types of 5G slices: eMBB and uRLLC. The mMTC slice can use the same algorithm proposed for the eMBB slice type. The proposed algorithms are used at the SO level and could be easily implemented in a real platform. The proposed algorithms used the CQI values provided between the eNB and the SO, to update the calculation of N_{pRB} dynamically.
The simulation results presented in Chapter 3, allow to verify the accuracy of our algorithms when estimating the needed pRBs for each type of slice. However, these results revealed two concerns, which we have addressed in the following contributions and which are:

- Channel Overload caused by CQI signaling between the eNB and the SO.

- The resource limit in the bandwidth that restricts the assignment of all the requested resources by the slice.

7.1.2 Channel stability prediction to optimize signaling overhead in 5G networks using ML

In the second contribution of this thesis, we focused on developing methods to reduce the signaling overhead caused by the periodic transmission of the CQI reports in 4G and 5G mobile networks. Therefore, we proposed a method based on ML algorithms. Our main approach consists of predicting the stability/mobility of channel conditions, in order to avoid transmitting the unnecessary CQI messages when the CQI values do not change significantly over time. The used ML algorithms are SVM and NN, and only require CQI information as input.
We compared, evaluated, and analyzed the prediction accuracy of the two ML schemes used for this purpose. We further addressed the trade-off between prediction accuracy and data collection frequency.

7.1.3 Data-driven RAN slicing mechanisms for 5G and beyond

In this contribution, we proposed two predictive methods to mitigate the overheads associated with frequent CQI monitoring, i.e., *Optimal Difference* and *LSTM*-based forecasting. In addition, to verify that the decreased collection of CQI values does not impact the N_{pRB} estimation, we applied this decreased CQI frequency to adjust the N_{pRB} needed by each running slice, in the algorithms that drive the N_{pRB} among heterogeneous slices.
Our objective is to minimize the error of our estimates of the achievable throughput when the CQI reporting frequency is reduced while reducing the signaling overhead between the eNB and the SO.
The simulation results proved that our solution is able to reduce the signaling overhead while maintaining a precise prediction of RAN resources for the running network slices.

7.1.4 On using reinforcement learning for network slice admission control in 5G: offline vs. online

In this contribution, we investigated the challenge of slice admission control. In this context, we focused on finding a policy at the InfProv to decide whether to accept or reject a new network slice request creation. The methods we used for this purpose are based on Reinforcement Learning algorithms (QL, DQL), and an online algorithm, which is Regret Matching. The problem is modeled as a MDP, and described how the proposed algorithms could derive a policy to be used by the agent to maximize the InfProv revenue while avoiding violating network slice requirements.
The simulation results showed that the proposed algorithms could derive a good

policy that maximize the InfProv revenues, while respecting the slice requirement in terms of resources. However, RM achieves the best performance, as the SAC problem is not complex and the actions state is limited.

7.2 Perspectives

Network slicing is a new technology, which introduces several new research directions. In this thesis, we have investigated some network slicing aspects in the RAN. Our contributions have verified their effectiveness in the proposed scenarios. However, there are still several scenarios that can integrate our solutions, as well as a future open issue in this context. In this section, we will present some perspectives of this thesis.

- **Two level admission control (UEs/slice and slices/InfProv)**

  Resource allocation among slices is a very critical issue, which opens wide perspectives for research. In the first contribution of this thesis, we proposed algorithms for resource sharing among slices. These algorithms allow estimating the N_{pRB} requested by each slice. However, the N_{pRB} available in the BW is limited and the SO cannot satisfy all the requested resources beyond the limit of the BW. In this context, if a slice receives lesser resources than it needs, it will not meet all the requirements of its hosted users. As a perspective, it will be interesting to integrate an admission control algorithm between the slice and the users who request to be hosted at a given slice. In fact, users run their applications in the slice in which they are served by their required resources. Therefore, if a slice does not receive all the resources necessary to meet all the requirements of the users who request to be hosted in this slice, it should reject some users by doing an admission control. Admission control between slice and users can be based on several constraints. This process improves the QoS requested by the users as well as by the slice.

  In Chapter 6, we presented algorithms of slice admission control between InfProv and slices. This issue deals only with the admission control between slices. Nevertheless, including a second level of admission control between users in each slice will strengthen the QoS of the whole network slicing.

- **Network slicing in the market**

  In this thesis, we introduced the concept of the slice market, aimed at maximising InfProv's revenues. In this context, the slice market is a critical concept that has to be considered in all technical solutions related to network slicing in general and resource allocation in particular. Therefore, we highlighted the

maximization of InfProv revenues using reinforcement learning. Therefore, our solution could be applied in various parts of the network, in the concept of slice market, such us to maximize operator revenues. In fact, reinforcement learning models the reward in a concrete way, ensuring maximum benefit to the corresponding network.

- **Multi-cell RAN resource allocation**

  The introduced resource allocation algorithms in this thesis, consider a single SO and eNB, a slice of type eMBB and of type uRLLC. The developed algorithm can be expanded to a multi-cell RAN. It can consider a problem of resource sharing over several eNBs, which in turn are shared by multiple slices of heterogeneous types. The requirements of each slice can be different, even if they are the same type.

- **Multi-cell RAN flexibility**

  Flexibility constraints become more critical in the case of multi-cell RAN. It can be addressed by ensuring that radio resources are allocated to its customers during a period i.e, a time slot. In this context, an in-depth study on how to correctly allocate a time slot during resource allocation needs to be carried out. These analyses can be based on ML classification algorithms or game theory.

- **Next Generation RAN**

  Our proposed solutions could be integrated into the Next Generation of RAN. In fact, our resource sharing algorithm is tested on a 4G network where the bandwidth contains only 25 RBs. However, our solution can be used in 5G networks with 100 RBs. In addition, our proposed algorithms could be integrated and tested easily in the new generation RAN in 5G networks.

- **AI driven RAN**

  Artificial intelligence (AI) has proven its effectiveness in several areas. Researchers in industry and academia have demonstrated the importance of implementing AI in the new generations of telecommunications. Indeed, RAN operations can be automated by integrating the AI into the RAN. Therefore, we are interested in providing AI-based solutions to improve RAN performance, especially in the new 5G radio. In this context, the problem of resource allocation between slices can be driven by a pure AI algorithm. For

instant: representing each slice by a class in an ML modeling, then classifying a given amount of resource on each slice for a given period of time according to each slice requirement.

Bibliography

[1] THALES, "Introducing 5g technology and networks (definition, use cases and rollout), white paper. available at: https://www.thalesgroup.com/en/markets/digital-identity-and-security/mobile/inspired/5g."

[2] Netmanias, "Lte network architecture: Basic," in *available at: https://www.netmanias.com/en/post/techdocs/5904/lte-network-architecture/lte-network-architecture-basic*, July 10, 2013.

[3] E. Dahlman, S. Parkvall, and J. Sköld, "4g lte/lte-advanced for mobile broadband," in *Academic Press, ISBN: 012385489X*, 2011, pp. 1–455.

[4] S. Niafar, X. Tan, and D. H. K. Tsang, "The optimal user scheduling for lte-a downlink with heterogeneous traffic types," in *10th International Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness*, 2014, pp. 56–62.

[5] CQI, in *LTE Quick Reference available at: https://www.sharetechnote.com/html/Handbook$_L$TE$_C$QI.html*.

[6] I. Farris, T. Taleb, Y. Khettab, and J. Song, "A survey on emerging sdn and nfv security mechanisms for iot systems," *IEEE Communications Surveys Tutorials*, vol. 21, no. 1, pp. 812–837, Firstquarter 2019.

[7] ETSI, "Network functions virtualisation (nfv); architectural framework," *GS NFV 002*, vol. 1.1.1, Oct. 2013.

[8] K. Rabie, "Core network evolution 5g service based architecture," *NETMANIASTECH-BLOG*, December 11, 2017.

[9] "5g nr gnb logical architecture and it's functional split options," *Available in: http://www.techplayon.com/5g-nr-gnb-logical-architecture-functional-split-options/*, December 11, 2017.

[10] I. Afolabi, T. Taleb, K. Samdanis, A. Ksentini, and H. Flinck, "Network slicing and softwarization: A survey on principles, enabling technologies, and

solutions," *IEEE Communications Surveys Tutorials*, vol. 20, no. 3, pp. 2429–2453, 2018.

[11] I. Afolabi, T. Taleb, P. A. Frangoudis, M. Bagaa, and A. Ksentini, "Network slicing-based customization of 5g mobile services," *IEEE Network*, vol. 33, no. 5, pp. 134–141, 2019.

[12] A. Ksentini, P. A. Frangoudis, A. PC, and N. Nikaein, "Providing low latency guarantees for slicing-ready 5g systems via two-level mac scheduling," *IEEE Network*, vol. 32, no. 6, pp. 116–123, 2018.

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

[14] T. Doukoglou, V. Gezerlis, K. Trichias, N. Kostopoulos, N. Vrakas, M. Bougioukos, and R. Legouable, "Vertical industries requirements analysis targeted kpis for advanced 5g trials," in *2019 European Conference on Networks and Communications (EuCNC)*, 2019, pp. 95–100.

[15] 3GPP, "Feasibility study on new services and markets technology enablers – stage 1," vol. TR 22.891 release 14.

[16] A. Ksentini and N. Nikaein, "Toward enforcing network slicing on ran: Flexibility and resources abstraction," *IEEE Communications Magazine*, vol. 55, no. 6, pp. 102–108, 2017.

[17] X. Li, M. Samaka, H. A. Chan, D. Bhamare, L. Gupta, C. Guo, and R. Jain, "Network slicing for 5g: Challenges and opportunities," *IEEE Internet Computing*, vol. 21, no. 5, pp. 20–27, 2017.

[18] S. Bakri, P. A. Frangoudis, and A. Ksentini, "Dynamic slicing of ran resources for heterogeneous coexisting 5g services," in *2019 IEEE Global Communications Conference (GLOBECOM)*, 2019, pp. 1–6.

[19] S. Bakri, M. Bouaziz, P. A. Frangoudis, and A. Ksentini, "Channel stability prediction to optimize signaling overhead in 5g networks using machine learning," in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, 2020, pp. 1–6.

[20] N. Tuan Le, M. A. Hossain, A. Islam, D. Kim, Y. J. Choi, and Y. M. Jang, "Survey of promising technologies for 5g networks," in *Hindawi Publishing Corporation Mobile Information Systems*, no. 2676589, 2016, pp. 1–25.

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

[22] N. Panwar, S. Sharma, and A. K. Singh, "A survey on 5g: the next generation of mobile communication," *Physical Communication*, vol. 18, pp. 64–84, 2016.

[23] A. Aissioui, A. Ksentini, A. M. Gueroui, and T. Taleb, "Toward elastic distributed sdn/nfv controller for 5g mobile cloud management systems," *IEEE Access*, vol. 3, pp. 2055–2064, 2015.

[24] H. Wu, L. Hamdi, and N. Mahe, "Tango: A flexible mobility-enabled architecture for online and offline mobile enterprise applications," in *Eleventh International Conference on Mobile Data Management*, 2010, pp. 230–238.

[25] 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.

[26] K. Benzekki, A. El Fergougui, and A. Elbelrhiti Elalaoui, "Software-defined networking (sdn): A survey," in *Security and Communication Networks*, vol. 18, no. 9, 2016, p. 5803–5833.

[27] "Sdn architecture," in *Open Netw. Found. Palo Alto, CA, USA*, 2014, pp. TR–502.

[28] C. Beckmann, "Réseaux : comprendre le sdn et le nfv," *LE DIGITAL TRANSFORME L'ENTREPRISE: available at: https://www.solutions-numeriques.com/reseaux-comprendre-le-sdn-et-le-nfv/*, vol. 21, 2016.

[29] B. Han, V. Gopalakrishnan, L. Ji, and S. Lee, "Network function virtualization: Challenges and opportunities for innovations," *IEEE Communications Magazine*, vol. 53, no. 2, pp. 90–97, 2015.

[30] The Next Generation Mobile Networks (NGMN) , "5g white paper," *NGMN Alliance, Frankfurt, Germany*, Feb. 2015.

[31] "Study on architecture for next generation system," *release 14, 3GPP, Sophia Antipolis, France, Rep. TR 23.799,*, Dec. 2016.

[32] "Framework of network virtualization for future networks, next generation network—future networks," *Int. Telecommun. Union, Geneva, Switzerland, ITU-T Recommendation Y.3011*, Jan. 2012.

[33] B. Cornaglia, G. Young, and A. Marchetta, "Fixed access network sharing," *Optical Fiber Technology*, vol. 26, pp. 2–11, Dec. 2015.

[34] A. D. La Oliva, X. C. Perez, A. Azcorra, A. D. Giglio, F. Cavaliere, D. Tiegelbekkers, J. Lessmann, T. Haustein, A. Mourad, and P. Iovanna, "Xhaul: toward an integrated fronthaul/backhaul architecture in 5g networks," *IEEE Wireless Communications*, vol. 22, no. 5, pp. 32–40, 2015.

[35] X. Li, R. Casellas, G. Landi, A. de la Oliva, X. Costa-Perez, A. Garcia-Saavedra, T. Deiss, L. Cominardi, and R. Vilalta, "5g-crosshaul network slicing: Enabling multi-tenancy in mobile transport networks," *IEEE Communications Magazine*, vol. 55, no. 8, pp. 128–137, 2017.

[36] A. S. Thyagaturu, Y. Dashti, and M. Reisslein, "Sdn-based smart gateways (sm-gws) for multi-operator small cell network management," *IEEE Transactions on Network and Service Management*, vol. 13, no. 4, pp. 740–753, 2016.

[37] Y. Zaki, Liang Zhao, C. Goerg, and A. Timm-Giel, "Lte wireless virtualization and spectrum management," in *WMNC2010*, 2010, pp. 1–6.

[38] R. Mahindra, M. A. Khojastepour, H. Zhang, and S. Rangarajan, "Radio access network sharing in cellular networks," in *2013 21st IEEE International Conference on Network Protocols (ICNP)*, 2013, pp. 1–10.

[39] R. Kokku, R. Mahindra, H. Zhang, and S. Rangarajan, "Nvs: A substrate for virtualizing wireless resources in cellular networks," *IEEE/ACM Transactions on Networking*, vol. 20, no. 5, pp. 1333–1346, 2012.

[40] T. Guo and R. Arnott, "Active lte ran sharing with partial resource reservation," in *2013 IEEE 78th Vehicular Technology Conference (VTC Fall)*, 2013, pp. 1–5.

[41] J. He and W. Song, "Appran: Application-oriented radio access network sharing in mobile networks," in *2015 IEEE International Conference on Communications (ICC)*, 2015, pp. 3788–3794.

[42] A. Gudipati, D. Perry, L. E. Li, and S. Katti, "Softran: Software defined radio access network," in *in Proc. ACM HotSDN, Hong Kong*, Aug. 2013, p. 25–30.

[43] X. F. N. Nikaein, M. M. Kassem, M. K. Marina, and K. Kontovasilis, "Flexran: A flexible and programmable platform for software-defined radio access networks," *Proc. ACM CoNEXT*, 2016.

[44] N. Nikaein, M. K. Marina, S. Manickam, A. W. Dawson, R. Knopp, and christian Bonnet, "Flexran: A flexible and programmable platform for software-defined radio access networks," *ACM SIGCOMM Computer Communication*, October 2014.

[45] N. Gkatzios, M. Anastasopoulos, A. Tzanakaki, and D. Simeonidou, "Compute resource disaggregation: An enabler for efficient 5g ran softwarisation," in *2018 European Conference on Networks and Communications (EuCNC)*, 2018, pp. 1–5.

[46] W. Wu, L. E. Li, A. Panda, and S. Shenker, "Pran: Programmable radio access networks," *Proc. ACM Hot Topics Netw*, p. 6, Los Angeles, CA, USA, Oct. 2014.

[47] M. Bansal, J. Mehlman, S. Katti, and P. Levis, "Openradio: A programmable wireless dataplane," *Proc. ACM HotSDN*, p. 109–114, Helsinki, Finland, Aug. 2012.

[48] "C-ran the road towards green ran white paper, version 2.5," in *Mobile Res. Inst. Beijing, China*, Oct. 2011.

[49] J. Wu, Z. Zhang, Y. Hong, and Y. Wen, "Cloud radio access network (c-ran): a primer," *IEEE Network*, vol. 29, no. 1, pp. 35–41, 2015.

[50] "Next generation fronthaul interfaces (ngfi)," *IEEE Standards association P1914. Accessed: Mar. 18, 2017. [Online]. Available: https://standards.ieee.org/develop/wg/NGFI.html*.

[51] P. Rost, C. Mannweiler, D. S. Michalopoulos, C. Sartori, V. Sciancalepore, N. Sastry, O. Holland, S. Tayade, B. Han, D. Bega, D. Aziz, and H. Bakker, "Network slicing to enable scalability and flexibility in 5g mobile networks," *IEEE Communications Magazine*, vol. 55, no. 5, pp. 72–79, 2017.

[52] "Study on management and orchestration of network slicing for next generation network," *3GPP Technical Specification TR 28.801*, Sept. 2017.

[53] A. Farrel, "Recent developments in service function chaining (sfc) and network slicing in backhaul and metro networks in support of 5g," in *2018 20th International Conference on Transparent Optical Networks (ICTON)*, 2018, pp. 1–4.

[54] S. Song and J. Chung, "Sliced nfv service chaining in mobile edge clouds," in *2017 19th Asia-Pacific Network Operations and Management Symposium (APNOMS)*, 2017, pp. 292–294.

[55] M. U. Alhuseini and M. M. Olama, "5g service value chain and network slicing framework using ecosystem modeling, agile delivery, and user-story automation," *IEEE Access*, vol. 7, pp. 110 856–110 873, 2019.

[56] A. M. Escolar, J. M. A. Calero, and Q. Wang, "Scalable software switch based service function chaining for 5g network slicing," in *2020 IEEE International Conference on Communications Workshops (ICC Workshops)*, 2020, pp. 1–6.

[57] T. Truong-Huu, P. Murali Mohan, and M. Gurusamy, "Service chain embedding for diversified 5g slices with virtual network function sharing," *IEEE Communications Letters*, vol. 23, no. 5, pp. 826–829, 2019.

[58] J. Pei, P. Hong, and D. Li, "Virtual network function selection and chaining based on deep learning in sdn and nfv-enabled networks," in *2018 IEEE International Conference on Communications Workshops (ICC Workshops)*, 2018, pp. 1–6.

[59] Z. Luo, C. Wu, Z. Li, and W. Zhou, "Scaling geo-distributed network function chains: A prediction and learning framework," *IEEE Journal on Selected Areas in Communications*, vol. 37, no. 8, pp. 1838–1850, 2019.

[60] S. I. Kim and H. S. Kim, "A research on dynamic service function chaining based on reinforcement learning using resource usage," in *2017 Ninth International Conference on Ubiquitous and Future Networks (ICUFN)*, 2017, pp. 582–586.

[61] H. Zhang, N. Liu, X. Chu, K. Long, A. Aghvami, and V. C. M. Leung, "Network slicing based 5g and future mobile networks: Mobility, resource management, and challenges," *IEEE Communications Magazine*, vol. 55, no. 8, pp. 138–145, 2017.

[62] R. Li, C. Wang, Z. Zhao, R. Guo, and H. Zhang, "The lstm-based advantage actor-critic learning for resource management in network slicing with user mobility," *IEEE Communications Letters*, vol. 24, no. 9, pp. 2005–2009, 2020.

[63] J. Heinonen, P. Korja, T. Partti, H. Flinck, and P. Pöyhönen, "Mobility management enhancements for 5g low latency services," in *2016 IEEE International Conference on Communications Workshops (ICC)*, 2016, pp. 68–73.

[64] A. S. D. Alfoudi, Q. O. Mosa, W. C. Alisawi, and R. N. Jaffer, "Slicing architecture for managing user mobility in next-generation heterogeneous wireless networks," in *2019 12th International Conference on Developments in eSystems Engineering (DeSE)*, 2019, pp. 879–882.

[65] A. A. Gebremariam, M. Chowdhury, M. Usman, A. Goldsmith, and F. Granelli, "Softslice: Policy-based dynamic spectrum slicing in 5g cellular networks," in *2018 IEEE International Conference on Communications (ICC)*, 2018, pp. 1–6.

[66] M. Srinivasan and C. S. R. Murthy, "Efficient spectrum slicing in 5g networks: An overlapping coalition formation approach," *IEEE Transactions on Mobile Computing*, vol. 19, no. 6, pp. 1299–1316, 2020.

[67] X. Li, K. Jiao, F. Jiang, J. Wang, and M. Pan, "A service-oriented spectrum-aware ran-slicing trading scheme under spectrum sharing," *IEEE Internet of Things Journal*, pp. 1–1, 2020.

[68] B. Khodapanah, A. Awada, I. Viering, J. Francis, M. Simsek, and G. P. Fettweis, "Radio resource management in context of network slicing: What is missing in existing mechanisms?" in *2019 IEEE Wireless Communications and Networking Conference (WCNC)*, 2019, pp. 1–7.

[69] R. Xie, J. Wu, R. Wang, and T. Huang, "A game theoretic approach for hierarchical caching resource sharing in 5g networks with virtualization," *China Communications*, vol. 16, no. 7, pp. 32–48, 2019.

[70] C. Marquez, M. Gramaglia, M. Fiore, A. Banchs, and X. Costa-Pérez, "Resource sharing efficiency in network slicing," *IEEE Transactions on Network and Service Management*, vol. 16, no. 3, pp. 909–923, 2019.

[71] A. S. D. Alfoudi, S. H. S. Newaz, A. Otebolaku, G. M. Lee, and R. Pereira, "An efficient resource management mechanism for network slicing in a lte network," *IEEE Access*, vol. 7, pp. 89 441–89 457, 2019.

[72] H. Yu, F. Musumeci, J. Zhang, M. Tornatore, and Y. Ji, "Isolation-aware 5g ran slice mapping over wdm metro-aggregation networks," *Journal of Lightwave Technology*, vol. 38, no. 6, pp. 1125–1137, 2020.

[73] X. Yang, Y. Liu, I. C. Wong, Y. Wang, and L. Cuthbert, "Effective isolation in dynamic network slicing," in *2019 IEEE Wireless Communications and Networking Conference (WCNC)*, 2019, pp. 1–6.

[74] N. Huin, P. Medagliani, S. Martin, J. Leguay, L. Shi, S. Cai, J. Xu, and H. Shi, "Hard-isolation for network slicing," in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, 2019, pp. 955–956.

[75] S. Vassilaras, L. Gkatzikis, N. Liakopoulos, I. N. Stiakogiannakis, M. Qi, L. Shi, L. Liu, M. Debbah, and G. S. Paschos, "The algorithmic aspects of network slicing," *IEEE Communications Magazine*, vol. 55, no. 8, pp. 112–119, 2017.

[76] P. Caballero, A. Banchs, G. de Veciana, and X. Costa-Pérez, "Multi-tenant radio access network slicing: Statistical multiplexing of spatial loads," *IEEE/ACM Transactions on Networking*, vol. 25, no. 5, pp. 3044–3058, 2017.

[77] A. Anand, G. De Veciana, and S. Shakkottai, "Joint scheduling of urllc and embb traffic in 5g wireless networks," in *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*, 2018, pp. 1970–1978.

[78] P. Caballero, A. Banchs, G. de Veciana, X. Costa-Pérez, and A. Azcorra, "Network slicing for guaranteed rate services: Admission control and resource allocation games," *IEEE Transactions on Wireless Communications*, vol. 17, no. 10, pp. 6419–6432, 2018.

[79] V. Sciancalepore, K. Samdanis, X. Costa-Perez, D. Bega, M. Gramaglia, and A. Banchs, "Mobile traffic forecasting for maximizing 5g network slicing resource utilization," in *IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*, 2017, pp. 1–9.

[80] D. Bega, A. Banchs, M. Gramaglia, X. Costa-Pérez, and P. Rost, "Cares: Computation-aware scheduling in virtualized radio access networks," *IEEE Transactions on Wireless Communications*, vol. 17, no. 12, pp. 7993–8006, 2018.

[81] J. X. Salvat, L. Zanzi, A. Garcia-Saavedra, V. Sciancalepore, and X. Costa-Pérez, "Overbooking network slices through yield-driven end-to-end orchestration," *Proc. ACM CoNEXT*, 2018.

[82] "Lte; evolved universal terrestrial radio access (e-utra); physical layer procedures," *3GPP TS 36.213, v. 15.2.0,*, vol. release 14.

[83] F. Capozzi, G. Piro, L. A. Grieco, G. Boggia, and P. Camarda, "Downlink packet scheduling in lte cellular networks: Key design issues and a survey," *IEEE Communications Surveys Tutorials*, vol. 15, no. 2, pp. 678–700, 2013.

[84] 3GPP, "Evolved universal terrestrial radio access (e-utra); user equipment (ue) radio transmission and reception," vol. TS 36.101 version 15.3.0 Release 15.

[85] M. Cordina and C. J. Debono, "A support vector machine based sub-band cqi feedback compression scheme for 3gpp lte systems," in *2017 International Symposium on Wireless Communication Systems (ISWCS)*, 2017, pp. 325–330.

[86] J. He and L. Sivridis, "A strategy to reduce the signaling requirements of cqi feedback schemes," in *Wireless Pers Commun*, vol. 70, May 2013, p. 85–98.

[87] M. Kang and K. S. Kim, "Performance analysis and optimization of best-m feedback for ofdma systems," *IEEE Communications Letters*, vol. 16, no. 10, pp. 1648–1651, 2012.

[88] M. Abdulhasan, M. Salman, and e. a. C.K. Ng, "An adaptive threshold feedback compression scheme based on channel quality indicator (cqi) in long term evolution (lte) system," *Wireless Pers Commun*, vol. 82(4), p. 2323–2349, Jun. 2015.

[89] A. Chiumento, M. Bennis, and C. D. et al, "Adaptive csi and feedback estimation in lte and beyond: a gaussian process regression approach," *EURASIP Journal on Wireless Communications and Networking*.

[90] R. A. Akl, S. Valentin, G. Wunder, and S. Stánczak, "Compensating for cqi aging by channel prediction: The lte downlink," in *2012 IEEE Global Communications Conference (GLOBECOM)*, 2012, pp. 4821–4827.

[91] M. A. Awal and L. Boukhatem, "Dynamic cqi resource allocation for ofdma systems," in *2011 IEEE Wireless Communications and Networking Conference*, 2011, pp. 19–24.

[92] O. Simeone, "A very brief introduction to machine learning with applications to communication systems," *IEEE Transactions on Cognitive Communications and Networking*, vol. 4, no. 4, pp. 648–664, 2018.

[93] I. El Emary, "On the application of artificial neural networks in analyzing and classifying the human chromosomes," *Journal of Computer Science*, vol. 2, no. 1, pp. 72–75, Jan. 2006.

[94] V. Kecman, "Support vector machines – an introduction," *Support Vector Machines: Theory and Applications. Studies in Fuzziness and Soft Computing*, vol. 177, pp. 1–47, Berlin, Heidelberg: Springer, 2005.

[95] I. Saffar, M. L. A. Morel, K. D. Singh, and C. Viho, "Machine learning with partially labeled data for indoor outdoor detection," in *2019 16th IEEE Annual Consumer Communications Networking Conference (CCNC)*, 2019, pp. 1–8.

[96] D. Powers, "Evaluation: From precision, recall and f-factor to roc, informedness, markedness correlation," vol. 2 (1), Jan. 2011, pp. 37–63.

[97] MATLAB Statistics and Machine Learning Toolbox, "Support vector machine classification," in *MathWorks*, 2019.

[98] Deep Learning Toolbox, "Formerly neural network toolbox," in *MathWorks*, 2019.

[99] MATLAB, "Long short-term memory (lstm)," in *Time Series Forecasting Using Deep Learning, MATLAB, Deep Learning Toolbox*.

[100] B. Koksal, R. Schmidt, X. Vasilakos, and N. Nikaien, "Crawdad dataset eurecom/elasticmon5g2019," vol. 2019-08-29, 2019.

[101] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. ICLR*, 2015.

[102] D. Bega, M. Gramaglia, A. Banchs, V. Sciancalepore, K. Samdanis, and X. Costa-Perez, "Optimising 5g infrastructure markets: The business of network slicing," in *IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*, 2017, pp. 1–9.

[103] M. Vincenzi, E. Lopez-Aguilera, and E. Garcia-Villegas, "Maximizing infrastructure providers' revenue through network slicing in 5g," *IEEE Access*, vol. 7, pp. 128 283–128 297, 2019.

[104] M. R. Raza, C. Natalino, P. Öhlen, L. Wosinska, and P. Monti, "Reinforcement learning for slicing in a 5g flexible ran," *Journal of Lightwave Technology*, vol. 37, no. 20, pp. 5161–5169, 2019.

[105] Y. Chen, Y. Gao, C. Jiang, and K. J. R. Liu, "Game theoretic markov decision processes for optimal decision making in social systems," in *2014 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, 2014, pp. 268–272.

[106] D. Nguyen, A. Rajagopalan, and C. Lim, "Online versus offline reinforcement learning for false target control against known threat," *Intelligent Robotics and Applications*, vol. 10985, 2018.