Technische Universität München

TUM School of Computation, Information and Technology

Design and Analysis of QoS and Network Slicing in Software-Defined Radio Access Networks

Arled Papa, M.Sc.

Vollständiger Abdruck der TUM School of Computation, Information and Technology der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften (Dr.-Ing.)

genehmigten Dissertation.

Vorsitzender:		Prof. Dr. Dr.-Ing. Holger Boche
Prüfer der Dissertation:	1.	Prof. Dr.-Ing. Wolfgang Kellerer
	2.	Assoc. Prof. Roberto Riggio

Die Dissertation wurde am 30.05.2022 bei der Technischen Universität München eingereicht und durch die TUM School of Computation, Information and Technology am 10.11.2022 angenommen.

Design and Analysis of QoS and Network Slicing in Software-Defined Radio Access Networks

Arled Papa, M.Sc.

10.11.2022

Abstract

It is anticipated that 5G networks will accommodate a wide range of applications with distinct Quality of Service (QoS) requirements. While initial mechanisms to enable such a realization in the core network side are consolidated in the 3rd Generation Partnership Project (3GPP) standardization, the Radio Access Network (RAN) counterpart is lacking concrete solutions.

Whereas RAN is envisioned to play a major role in the 5G era, current rigid and monolithic RAN implementations cannot support the demanded level of heterogeneity and flexibility in the network. This has led to the emergence of solutions such as Software-Defined Networking (SDN) and network slicing. The former is envisioned as a tool to foster management and orchestration in RAN, paving the way towards a paradigm shift referred to as Software-Defined Radio Access Network (SD-RAN). The latter is foreseen as a mechanism to enable the coexistence of multiple applications within the same infrastructure, but operating in an isolated fashion from one another. SD-RAN triggers programmability and flexibility by separating the data plane and control plane of RAN, transferring control to centralized entities known as SD-RAN controllers. However, the concept of SD-RAN standalone cannot fulfill the stringent 5G requirements. To that end, a combination with the features provided from network slicing becomes necessary. Yet, the development of efficient approaches that leverage characteristics from both aforementioned technologies is a non-trivial task and constitutes a set of challenges. These challenges mainly concern the establishment of effective architectural concepts and development of optimization solutions that improve network efficiency.

In this thesis, in principle, the integration of SD-RAN and RAN slicing within the existing 5G infrastructure is investigated. This is achieved following the concepts of adaptability and flexibility as well as control and data plane separation offered in terms of a service-based architecture. In that regard, interfaces and solutions to realize such a merge in an efficient manner are provided.

While RAN slicing allows the coexistence of applications with distinct QoS requirements in the network, the development of an effective infrastructure that leverages the SD-RAN features and provides efficient optimization solutions opens up a wide area of research. In this thesis, initially, an architectural concept for RAN slicing based on the SD-RAN paradigm is

proposed, where the wireless resource allocation is performed in two steps: firstly on the SD-RAN controller, referred to as **inter-slice scheduler**, and secondly within the Base Station (BS), known as **intra-slice scheduler**. This method triggers effective network decisions originating mainly from maintaining a broader network view via the SD-RAN controllers. Further, while providing a guaranteed performance to individual users, this thesis develops and investigates RAN slicing optimization approaches based on stochastic optimization tools both for single and multi-BS scenarios. Results demonstrate that the designed algorithms improve network performance in terms of user throughput compared to state-of-the-art approaches while always satisfying users' QoS.

In such a setup, where RAN control is centralized in entities known as SD-RAN controllers, and considering the drastic increase on the number of network devices with applications like Internet-of-Things (IoT), such controllers become a potential bottleneck. Therefore, their performance may lead to unpredictable behavior and can result in network inconsistencies. In this thesis, a tool is proposed, called MARC, which is able to model and evaluate the predictability of state-of-the-art SD-RAN controllers in the control plane.

However, whereas providing predictability for SD-RAN controllers in the control plane is a first step towards optimality, the effect on the data plane, where the users lie is of utmost importance. Consequently, in this thesis, two main contributions in that regard are provided. The first lies in the predictability of the throughput performance of each user with respect to various resource allocation policies in the network. This is achieved via theoretical models, simulations, and measurements. The second contribution concerns the modeling and development of a 5G-based SD-RAN simulator that investigates the possibility of a distributed control plane. Moreover, it derives important conclusions with respect to user performance in the data plane while comparing the centralized and distributed control plane approaches.

The wide range of 5G applications include scenarios beyond typical network infrastructures, for instance incorporating non-terrestrial and aeronautical networks. This leads to the emergence of applications such as In-Flight Entertainment and Connectivity Services (IFECS). In order to support ubiquity in 5G, a smooth operation between the above mentioned networks and the existing terrestrial network must be maintained. Thus, similarly to terrestrial networks, SDN and network slicing are leveraged for these **multi-network environments**. In this thesis, an infrastructure design for IFECS based on SD-RAN and network slicing, considering resources beyond RAN, is introduced. This includes satellite links and caching resources, which are incorporated into what is called the **multi-network slicing** problem. Further, a new definition of network slicing is established that is based on the willingness of Service Providers (SPs) to reveal information about their customers. According to this definition, results with respect to the cost of a network slice are demystified.

Kurzfassung

5G-Netze werden voraussichtlich ein breites Spektrum von Anwendungen mit unterschiedlichen Anforderungen an die Quality of Service (QoS) beinhalten. Während sich erste Mechanismen zur Ermöglichung eines solchen Konzepts auf der Core-Network-Seite in der Standardisierung des Third Generation Partnership Project (3GPP) verfestigen, fehlt es auf der Radio Access Network (RAN)-Gegenseite an konkreten Lösungen.

Wenngleich das RAN in der 5G-Ära eine wichtige Rolle spielen soll, können die derzeitigen starren und monolithischen RAN-Implementierungen nicht das geforderte Maß an Heterogenität und Flexibilität im Netzwerk unterstützen. Dies führt schließlich zum Aufkommen von Konzepten wie Software-Defined Networking (SDN) und Network Slicing. Ersteres wird als Instrument zur Entwicklung von Management und Orchestrierung im RAN betrachtet, was zu einem Paradigmenwechsel führt, welcher als Software-Defined Radio Access Network (SD-RAN) bezeichnet wird. Letzteres ist als Mechanismus vorgesehen, der die Koexistenz mehrerer Anwendungen innerhalb derselben Infrastruktur ermöglichen soll, die isoliert voneinander betrieben werden. SD-RAN schafft Programmierbarkeit und Flexibilität durch die Trennung von Datenebene und Steuerungsebene des RAN und überträgt die Steuerung von Einheiten zentralisiert an sogenannte SD-RAN-Controller. Diese Maßnahme allein kann jedoch die strengen 5G-Anforderungen nicht erfüllen. Daher ist eine Kombination mit den von Network Slicing bereitgestellten Funktionen erforderlich. Die Entwicklung effizienter Ansätze, welche die Eigenschaften der beiden vorgenannten Technologien nutzen, ist nicht trivial und stellt die Forschungswelt vor einige Herausforderungen. Diese betreffen vor allem die Erstellung effektiver Architekturkonzepte und die Entwicklung von Optimierungslösungen.

In dieser Dissertation wird die Integration von SD-RAN- und Network Slicing-Konzepten in die bestehende 5G-Architekturinfrastruktur untersucht, wobei der Fokus darauf liegt, Anpassungsfähigkeit und Flexibilität in den Vordergrund zu rücken, ebenso wie die Abstraktion von Steuerungs- und Datenebene in einer service-basierten Architektur. In diesem Zusammenhang werden Schnittstellen und Instrumente zur Verfügung gestellt, um eine solche Verschmelzung auf effiziente Weise zu realisieren.

Während RAN Slicing die Koexistenz von Anwendungen mit unterschiedlicher QoS im Netzwerk bietet, eröffnet die Entwicklung einer effektiven Infrastruktur – welche die Möglichkeiten der SD-RAN mit effizienten Optimierungslösungen verbindet – ein weites Feld für die Forschung. In dieser Dissertation wird zunächst ein architektonischer Ansatz für RAN-Slicing vorgestellt, der auf dem SD-RAN-Paradigma beruht, bei dem die Zuweisung von Wireless-Ressourcen in zwei Schritten erfolgt: Erstens im SD-RAN-Controller, dem sogenannten **inter-slice** scheduler, und zweitens innerhalb der Basisstation, dem sogenannten **intra-slice** scheduler. Diese Methode führt zu effektiven Netzwerkentscheidungen, vor allem, da SD-RAN-Controller somit einen umfassenderen Überblick über das Netzwerk erhalten. Während eine garantierte Leistung für Benutzer sichergestellt wird, werden RAN-Slicing-Optimierungsansätze auf der Grundlage stochastischer Optimierungswerkzeuge sowohl für Szenarien mit einer als auch mit mehreren Basisstationen untersucht. Die Ergebnisse zeigen, dass die entwickelten Algorithmen die Netzwerkleistung bezüglich der Benutzerleistung im Vergleich zu state-of-the-art Ansätzen verbessern und die QoS der Benutzer erfüllen.

In einer solchen Konfiguration, in der die RAN-Steuerung zentral in SD-RAN-Controllern erfolgt, werden solche Controller zu einem Engpass im Netzwerk, insbesondere angesichts der drastischen Zunahme an Netzwerkgeräten mit dem Aufkommen von Applikationen wie dem Internet-of-Things (IoT). Daher kann es zu unvorhersehbarem Verhalten und zu Netzwerkinkonsistenzen in ihrer Leistung kommen. Diese Dissertation stellt ein Tool namens MARC vor, welches die Steuerungsebene von state-of-the-art SD-RAN-Controllern modelliert und somit ihr Verhalten vorhersagbar macht.

Während die Sicherstellung der Berechenbarkeit des Verhaltens von SD-RAN-Controllern in der Steuerungsebene ein erster Schritt in Richtung des Optimums ist, sind die Auswirkungen auf die Datenebene, auf der sich die Benutzer befinden, indes von größter Bedeutung. Folglich werden in dieser Dissertation zwei wesentliche Beiträge diesbezüglich vorgelegt: Der Erste liegt dabei in der Vorhersagbarkeit der Durchsatzleistung bei den einzelnen Benutzern in Anbetracht der verschiedenen Ressourcenzuweisungsstrategien im Netzwerk. Dies wird sowohl mit Simulationen, Messungen, als auch mit theoretischen Modellen nachgewiesen. Der zweite Beitrag betrifft die Simulation und Entwicklung eines auf 5G SD-RAN gestützten Simulators, der die Perspektive einer verteilten Steuerungsebene untersucht. Darüber hinaus werden wichtige Erkenntnisse in Bezug auf die Leistung und Produktivität von Benutzern in der Datenebene abgeleitet und die Herangehensweisen der zentralisierten und verteilten Steuerungsebenen verglichen.

Das breite Spektrum der 5G-Anwendungen umfasst Szenarien außerhalb typischer Netzwerkinfrastrukturen, wie beispielsweise die Einbindung satelliten-gestützter und flugtechnischer Netze. Diese ermöglichen das Entstehen von Anwendungen für Satelliten, In-Flight

Entertainment und Connectivity Services (IFECS). Um die flächendeckende Verbreitung von 5G zu unterstützen, muss ein reibungsloser Betrieb in den ebengenannten Netzen mit dem bestehenden terrestrischen Netzwerk sichergestellt werden. Ähnlich wie bei terrestrischen Netzwerken, werden SDN und Network Slicing für diese **Multi-Network Environments** nun ebenfalls wirksam eingesetzt. In dieser Dissertation wird ein Infrastrukturdesign für IFECS-Systeme auf der Grundlage von SD-RAN und Network Slicing sowie Ressourcen oberhalb des RANs vorgestellt. Dieses beinhaltet Satellitenverbindungen und Caching-Speicher, was als **Multi-Network Slicing** bezeichnet wird. Zudem wird eine neue Definition für Network Slicing vorgestellt, auf Basis der Bereitschaft von Service Providern, Informationen über ihre Kunden preiszugeben. Auf Grundlage dieser Definition werden Erkenntnisse bezüglich der Kosten von Network Slicing entmystifiziert.

Acknowledgment

The last five years in the course of pursuing my doctorate title have left an important mark and have played an immense role in developing the individual that I am today, both academically and personally. Looking back at those memories, it will always remind me of the challenges, the ups and downs, but most importantly the incredible people with whom I had the opportunity to collaborate, interact and go through this journey together. Although words cannot be enough to describe the feelings and emotions of all those years, I would still like to try to express my gratitude to all of those who accompanied me in the achievement of this milestone in my life.

First, I want to thank Prof. Wolfgang Kellerer for providing me this once in a life time opportunity to be part of this unique family at LKN and to be able to fulfill my doctorate research at such a prestigious university. I was lucky to be part of exciting projects and be able to work with equipment and testbeds that even industrial entities would dream of working with. Furthermore, I want to thank professor for his guidance and support that helped me overcome all the challenges and achieve the fulfillment of my thesis.

While working hard is the core of success in life, nothing would be possible without support and guidance from people in your closest environment. Thus, I am very thankful to all my former colleagues in LKN for making work such a nice place to be at. In particular, I want to thank Dr.-Ing. Murat Gürsu and Dr.-Ing. Raphael Dürner for sharing the office for many years with me, guiding me and for enduring my countless questions and talks every day for the first 3 years. Additionally, I want to thank Dr.-Ing. Markus Klügel and Dr. Fidan Mehmeti for the many discussions, the guidance and their great support. For the many interesting discussions and for all the relaxing evenings at the chair and events that helped me unwind and forget the daily stress and routine I want to specially thank Endri Goshi, Onur Ayan, Sai Kireet Patri, Nemanja Ðerić, Dr.-Ing. Amir Varastehhajipour, Dr.-Ing. Amaury Van Bemten, Dr.-Ing. Mikhail Wilhelm, Dr.-Ing. Christian Sieber, Serkut Ayvaşık and Yalgiz Özgan. Finally, great appreciation and thanks are attributed to Alba Jano and Polina Kutsevol for their contributions in their Master's thesis that lead to contents presented in this thesis.

While everything mentioned above are a result of hard work and constant support from friends and colleagues, none of those would have existed without the help and determination

of my family. It was them who with their constant support and sacrifices in raising me led to the person I am today. Leaving home for a foreign country at a young age was tough, however their unconditional love and courage made me fight through all difficulties. Moreover, I want to express my gratitude to my girlfriend Stefanie for her great support towards the challenging last years of my doctorate. Her positive energy, strong character and much appreciated patience eased the finalization of this journey.

Last but not least, I want to express my deepest appreciation for my friends that are constantly there for me. In particular I want to thank Elvis Majushi, Miodrag Davidović, Ece Öztürk, Alex Git and Kristian Kuro for helping me regenerate and enjoy moments outside my daily routine and research.

Contents

Acronyms

3GPP 3rd Generation Partnership Project 3, 18, 23, 24, 55, 56, 58, 60, 61, 63, 79, 80, 82, 94, 97, 98, 119, 129, 145

5GC 5G Core 3, 11, 12, 14, 18, 21, 55, 56, 57, 58, 59, 60, 61, 145

5GS 5G System Architecture 11, 12

AF Application Function 13, 56, 58, 60

AMF Authentication Mobility Function 12, 14, 56, 58, 61

AoI Age of Information 1

API Application Programmable Interface 16, 29, 43, 54, 58, 59, 60, 67, 70, 89, 98

AUSF Authentication Server Function 13

BS Base Station 2, 3, 4, 7, 8, 9, 14, 17, 18, 19, 20, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 41, 42, 43, 44, 47, 48, 49, 50, 51, 52, 53, 54, 55, 62, 63, 64, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 78, 79, 81, 82, 83, 84, 85, 86, 87, 88, 89, 91, 92, 93, 95, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 116, 117, 118, 121, 122, 128, 146, 169, 170

CCDF Complementary Cumulative Distribution Function 105

CN Core Network 2, 31

CP Control Plane 13, 57

CPU Central Processing Unit 7, 19, 42, 53, 54, 61, 73, 75, 81, 82, 83, 84, 85, 86, 88, 110, 116, 117, 118, 121, 122, 125, 126, 127, 128, 170

CQI Channel Quality Indicator 29, 30, 97, 98, 99, 100, 101, 102, 103, 104, 106, 109, 110, 112, 113, 115, 119, 120, 121, 122

CU Central Unit 57

CU-UP Central Unit User Plane 18, 66

CU-CP Central Unit Control Plane 18, 66

DA2G Direct Air-to-Ground 131

DN Data Network 12, 59

DRB Data Radio Bearer 14, 57

DU Distributed Unit 18, 56, 57, 66

eDECOR Enhanced Dedicated Core 18

eMBB Enhanced Mobile Broadband 1, 79, 80

eNB Evolved NodeB 95, 115, 116

EPA Extended Pedestrian A 41, 42

GEO Geostationary Equatorial Orbit 131

gNB Next Generation NodeB 2, 13, 14, 17, 18, 55, 56, 57, 61, 95, 97, 115, 116, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128

HAP High-Altitude Platform 143

HSS Home Subscriber Server 12, 97

IFECS In-Flight Entertainment and Connectivity Services 8, 131, 132, 134, 144, 147, 148

InP Infrastructure Provider 25, 134

IoT Internet-of-Things 1, 63, 64

IP Internet Protocol 12, 14, 18, 87, 97, 99

ITU International Telecommunications Union 1

JSON Javascript Object Notation 59

KPI Key Performance Indicator 61

LC Logical Channel 69, 70, 72, 79

LiFi Light Fidelity 133, 134

LTE Long Term Evolution 1, 11, 15, 25, 41

MAC Medium Access Control 4, 8, 14, 17, 21, 57, 91, 98, 99, 100, 101, 119, 120, 122

maxCQI maximum Channel Quality Indicator 92, 93, 100, 103, 107, 109, 110, 111, 112, 113, 114, 115, 120, 121, 123, 124, 128

MCS Modulation and Coding Scheme 103

MEC Mobile Edge Computing 63, 74, 148

MILP Mixed-Integer Linear Programming 9

MINLP Mixed-Integer Nonlinear Programming 132, 144

MLF Most Linked First 47, 48, 52, 53, 54

MME Mobility Management Entity 12, 97

mMTC Massive Machine-Type Communications 1, 80

NBI Northbound Interface 16

near-RT RIC Near-Real Time RAN Intelligent Controller 18, 67, 68, 95, 115

NEF Network Exposure Function 13

NF Network Function 3, 13, 58

NG Next Generation 61

non-RT RIC non-Real Time RAN Intelligent Controller 18, 95

NR New Radio 3, 4, 11, 12, 13, 14, 15, 21, 55, 56, 58, 59, 60, 145

NRF Network Repository Function 13

NRM Network Resource Model 59

NSI Network Slice Instance 58, 59, 60

NSSF Network Slicing Selection Function 13, 61

NSSI Network Slice Subnet Instance 58, 59, 60

OFDM Orthogonal Frequency Division Multiplexing 15

PC Personal Computer 81, 82, 87, 110

PCF Policy Control Function 13

PCRF Policy Charging Rules Function 12

PDCP Packet Data Convergence Protocol 14, 56, 57

PDU Protocol Data Unit 61

PGW Packet Gateway 12, 97

PHY Physical Layer 14, 57

PMF Probability Mass Function 100, 101, 103, 104, 110, 132

PRB Physical Resource Block 14, 15, 20, 21, 25, 26, 27, 33, 36, 37, 38, 42, 43, 44, 46, 47, 50, 54, 98, 100, 101, 102, 103, 104, 105, 106, 108, 109, 119, 121, 122, 124, 125, 127, 128, 169

QoS Quality of Service 2, 5, 6, 7, 9, 11, 14, 18, 19, 21, 30, 31, 32, 36, 37, 39, 40, 41, 43, 47, 48, 50, 53, 57, 58, 59, 89, 93, 115, 118, 124, 125, 126, 129, 133, 143, 145, 146, 148

RAM Random Access Memory 28, 82, 110

RAN Radio Access Network 2, 3, 4, 5, 6, 7, 8, 11, 13, 14, 15, 17, 18, 19, 20, 21, 23, 24, 25, 28, 29, 30, 31, 40, 41, 42, 43, 44, 45, 46, 48, 49, 50, 55, 56, 57, 58, 59, 60, 61, 62, 63, 64, 65, 66, 67, 70, 71, 72, 73, 74, 80, 84, 89, 91, 92, 93, 94, 95, 96, 98, 101, 119, 122, 131, 132, 133, 134, 143, 145, 146, 147

RANCF Radio Access Network Control Function 55, 56, 57, 58, 59, 60, 61, 62

RE Resource Element 15

REST Representational State Transfer 16, 58, 59, 60, 62, 67, 70

RIB RAN Information Base 70, 81, 83, 85, 86, 89, 115, 116, 117, 120

RIC RAN Intelligent Controller 95

RLC Radio Link Control 14, 56, 57

RR Round-Robin 48, 92, 93, 99, 100, 103, 104, 109, 110, 111, 112, 113, 114, 115, 128

RRC Radio Resource Control 14, 17, 18, 57

RU Radio Unit 18, 66

SBI Southbound Interface 16

SDAP Service Data Application Protocol 14

SD-RAN Software-Defined Radio Access Network 2, 3, 4, 5, 6, 7, 8, 9, 11, 17, 18, 19, 20, 21, 23, 24, 25, 28, 29, 32, 36, 43, 54, 55, 56, 57, 58, 59, 62, 63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83, 84, 85, 87, 88, 89, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 109, 110, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 125, 126, 128, 129, 134, 135, 136, 137, 141, 142, 143, 145, 146, 147, 148, 173

SDN Software-Defined Networking 2, 3, 5, 11, 15, 16, 17, 19, 21, 23, 28, 54, 55, 58, 59, 63, 64, 66, 72, 73, 81, 91, 96, 132, 143, 145, 147, 148

SGW Service Gateway 12, 97

SINR Signal-to-Interference-Plus-Noise-Ratio 41, 42, 48, 53, 54, 103

SMF Session Management Function 12, 56, 58, 59, 60, 61

SP Service Provider 8, 9, 132, 134

SRB Signal Radio Bearer 14, 57

TCP Transport Control Protocol 18, 66, 77, 82, 87, 98, 101, 115, 118, 125

TTI Transmission Time Interval 25, 119, 121, 123, 128

UAV Unmanned Aerial Vehicles 143

UDM Unified Data Management 12

UE User Equipment 4, 5, 12, 14, 18, 20, 21, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 36, 38, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 58, 59, 60, 61, 62, 63, 65, 66, 69, 70, 71, 72, 73, 74, 75, 76, 78, 79, 80, 81, 82, 83, 84, 85, 86, 87, 88, 89, 91, 92, 93, 94, 95, 96, 97, 98, 99, 100, 101, 102, 103, 104, 105, 106, 107, 108, 109, 110, 111, 112, 113, 114, 115, 116, 117, 118, 119, 120, 121, 122, 123, 124, 125, 126, 127, 128, 129, 169, 170

UP User Plane 13, 57

UPF User Plane Function 12, 58, 59, 61

URLLC Ultra-Reliable Low-Latency Communications 1, 15, 80

VMNO Virtual Mobile Network Operator 2, 7, 20, 23, 25, 29, 31, 41, 43

VoIP Voice Over-IP 131

WiFi Wireless Fidelity 134

XML Extensible Markup Language 59

Chapter 1

Introduction

Communication networks have become an important aspect of our digital society, revolutionizing the way we communicate starting from the early 1990s. Yet, the focus of this digital transformation has been in constant change over the last couple of years. Whereas originally designed for voice (2G) and data communication (3G), mainly embracing best-effort human traffic, a shift towards predictable and high data rate communication has been observed with the development of 4G, Long Term Evolution (LTE) networks. This has led to the emergence of new techniques in network management and orchestration, improving spectral efficiency and providing support for high mobility scenarios. However, existing applications generally have assumed homogeneous requirements adequate to be served by a monolithic network design.

While a previously considered *one-size-fits-all* network infrastructure has been sufficient up to the 4G era, emerging 5G networks have abandoned this assumption, in order to be able to provide service to very stringent-requirement types of traffic. Indeed, International Telecommunications Union (ITU) has defined three broad use cases for 5G networks, namely Enhanced Mobile Broadband (eMBB), Ultra-Reliable Low-Latency Communications (URLLC) and Massive Machine-Type Communications (mMTC) [Ser15]. The aforementioned applications need to coexist in the same infrastructure, but simultaneously preserve their distinct requirements. For instance, eMBB traffic type envisions 100 Mbps downlink peak data rates, URLLC applications aim at maintaining network delay below 1 ms, whereas the mMTC traffic class should accommodate up to 10^6 devices/km^2 [ITU]. Consequently, emerging network infrastructures need to cater for this level of heterogeneity. Therefore, new management and orchestration concepts need to be developed.

To foster improved management and orchestration techniques for emerging applications and research topics such as Internet-of-Things (IoT) [Oug+19], Age of Information (AoI) [Aya+20b; AÖK21], e-health [LEH20], tele-operations as well as augmented reality [Gup+19], a new softwarization approach referred to as Software-Defined Networking (SDN) [McK+08] has been developed. SDN envisions a separation of the control and data plane of traditional networks, where control is shifted towards centralized entities known as SDN controllers in the form of software functions. These functions are easily re-programmable and adaptable, bringing enhanced flexibility into mobile networks [Xia+14]. While SDN provides a step towards optimality in networking, it cannot cater standalone for all the required level of network flexibility. To overcome this issue, a new emerging approach known as network slicing is proposed in 5G [All16]. Network slicing allows for a logical network infrastructure separation, where multiple applications with distinct Quality of Service (QoS) requirements can be maintained. In this case, logical chunks of the network are separated into what are called **network slices** and associated to a group of users or applications, which are served by Virtual Mobile Network Operator (VMNO)s referred to as **tenants**. However, while network slicing provides a solution to the heterogeneity issue, it renders network slice orchestration harder, since it must ensure full isolation among network slices. We refer to **slice isolation** as a concept that defines the ability of each network slice to fulfill its requirements without being affected by other network slices.

Merging the paradigms of SDN and network slicing allows for a combination of features of both: a flexible and dynamic resource mapping to applications through network slicing, and a programmable and adaptable orchestration of applications with SDN. However, whereas the concepts of SDN and network slicing have been widely used in the Core Network (CN) [Jin+13; Ble+15; Qaz+17; Afo+18; Mar+18], only recently they have spurred into the Radio Access Network (RAN) counterpart. This has led to the emergence of Software-Defined Radio Access Network (SD-RAN), where now the control part of the RAN is centralized in what are called SD-RAN controllers. This brings a lot of benefits into the mobile networks [Fou+16; CKR19; Pap+20b], since it detaches the monolithic RAN control and enables cooperation among RAN components (i.e., Base Station (BS)s, known as Next Generation NodeB (gNB)s in 5G) and empowers orchestration of RAN functionalities. However, many important and interesting questions arise with respect to the functionality, integration and operation of SD-RAN in 5G.

In this thesis, we tackle specifically the problem of integrating SD-RAN into the 5G network architecture. Moreover, we investigate solutions that merge SD-RAN and network slicing to make the guaranteed QoS an inherent feature of the cellular network. To achieve this, first we gain a deeper understanding of the existing SD-RAN solutions in the literature and provide insights on performance guarantees, which is a prerequisite to construct new models and architectures. Then, as QoS is strongly tied to efficient resource allocation, we develop algorithms to boost network performance, while simultaneously utilizing RAN resources effectively and efficiently. Finally, to achieve network ubiquity and to pave the way towards new emerging use cases, we design systems that allow for network slicing even in multi-network environments that include not only cellular networks, but also satellite and aeronautical networks. The remainder of this chapter is organized as follows. Section 1.1 highlights the main research questions and challenges that arise within the concepts of SD-RAN and network slicing, which are tackled in this thesis. Further, Section 1.2 elaborates on the contributions of this thesis with respect to the raised challenges. Finally, Section 1.3 describes the outline of the thesis.

1.1 Research Questions

The design, optimization and implementation of SDN and network slicing concepts in RAN consists of various challenges. This section highlights the main research challenges investigated in more detail in Chapters 3 to 6 of this thesis and summarizes the main contributions.

Integration of SD-RAN into the existing 5G network infrastructure

5G networks envision programmability and softwarization as the major components to cater for the heterogeneous application requirements and to achieve adaptable and efficient network management [O-Ra]. More specifically, the novelty of 5G manifests in decoupling the control plane from data plane as well as in the abstraction of the control plane functions in terms of a service-based architecture [3GPa]. However, while the 5G Core (5GC) and New Radio (NR) functions are well defined into 3rd Generation Partnership Project (3GPP) standardization [3GPb; 3GPd], novel concepts such as SD-RAN still remain unspecified in the standard.

The introduction of SD-RAN into standardization needs to consider many aspects. First, in order to fit into the concept of a service-based architecture, the SD-RAN functionalities should be offered in terms of Network Function (NF)s. This requires clear definitions of the envisioned functionalities, as well as methods on how to retrieve them. Furthermore, methods and protocols should be developed for the interaction of the SD-RAN functions with the already existing 5G infrastructure components. Finally, concrete use cases need to be identified to establish a clear necessity for the newly introduced concept.

Design and Optimization of Efficient RAN Slicing Algorithms and Architectures

The introduction of SD-RAN enables programmability and fosters flexibility for 5G networks while alleviating network management and orchestration of BSs. However, the softwarization concept standalone cannot cater for the heterogeneous requirements of emerging 5G applications. In that regard, network slicing is envisioned as a tool to cater for such requirements of the mobile community [Mar+16]. Yet, RAN slicing has not been vastly studied in the literature. Consequently, many concepts including infrastructure and architecture design with respect to network slicing, clear definition of slice isolation as well as efficient slice resource allocation based on a per user level are limited.

Initially, the main question lies in the clear definition of RAN slicing and the differentiation from traditional Medium Access Control (MAC) scheduling approaches. Besides providing a concrete differentiation for the above mentioned concepts, the introduction of **slice isolation** is necessary. Generally speaking, slice isolation is often perceived as the ability of individual **network slices** to coexist, without interfering to each other's state of operation. However, in RAN, maintaining the required level of isolation is a non-trivial task, mainly due to the stochastic nature of the wireless channels, characterized by a high variability and the complex resource coupling among network slices.

Furthermore, to achieve an optimal wireless resource allocation, optimization techniques that take into account the specifics of wireless characteristics need to be considered. Static solutions that assume a unique and unchanged resource allocation do not suffice due to the constantly changing nature of wireless channels. Therefore, methods that envision and enable dynamic reconfigurations become mandatory.

Such a solution is enabled by the concept of SD-RAN. However, the combination of SD-RAN and network slicing renders the network management and network slice orchestration very complex. Consequently, the development of new network infrastructures and architecture concepts that bind these two approaches are of utmost importance.

Measurement and Analysis of SD-RAN platforms towards predictable QoS provisioning

The adoption of SD-RAN into the mobile community envisions the separation of the control plane and data plane of NR. Traditional RAN components such as BSs and User Equipment (UE)s reside in the data plane, whereas the control plane is centralized in entities referred to as SD-RAN controllers. However, the envisioned centralized approach becomes a potential performance bottleneck [Pap+21a].

When network dimensions increase (i.e., increasing the number of BSs, UEs), the control plane messages processed by the SD-RAN controllers also increase. Consequently, the SD-RAN controller operation becomes unpredictable and may lead to network anomalies. While state-of-the-art SD-RAN solutions provide initial insights into controller performance under a low number of RAN components (e.g., below 30) [Fou+16; CKR19; SIN21], large-scale and in-depth performance evaluations are missing and therefore the predictability of SD-RAN controllers remains unknown.

A general evaluation approach needs to cater for many aspects. However, the design of a tool that is able to scale to realistic 5G network dimensions is challenging. On the one hand, current open-source SD-RAN controllers are implementation-specific and they can only be operated utilizing expensive wireless hardware [Ett], which makes the extension of testing platforms infeasible. Hence, scaling the network would require a tremendous investment in space and time for research and testing purposes. On the other hand, the SD-RAN paradigm is still not standardized. Therefore, the extensibility towards new SD-RAN controllers without changing the system itself requires an intensive study of the design and implementation of SD-RAN benchmarks. In a similar fashion, capturing the effect of SD-RAN controllers on the UEs requires application-specific data plane modeling and the creation of efficient tools for measurements and experimentation. Whereas centralized SD-RAN controller solutions may become a potential bottleneck for the network, the introduction of a distributed control plane becomes necessary. However, the benefits and drawbacks of such an approach remain unknown and need to be validated with the development of tools based on realistic SD-RAN features.

Design and Analysis of Network Slicing Solutions for Multi-Network Environments

5G and beyond networks, envision network ubiquity. As a result, the network infrastructure should also cater for covering multi-network environments. In this thesis, we will consider multi-network environments that include not only cellular networks, but also non-terrestrial networks such as satellites and aeronautical networks (i.e., aircrafts). Whereas the integration of such multi-network environments into the current 5G infrastructure is hard, the problem becomes even more challenging considering the distinct application requirements of 5G/6G networks. Therefore, SDN and network slicing are anticipated as viable solutions to enable this integration. However, the existing solutions in the literature do not provide a method to achieve the aforementioned goal. Therefore, performing a detailed analysis of the existing system is required. To that end, new architecture designs combining SDN and network slicing need to be created to assess the overall network performance.

1.2 Contributions

This section summarizes the main contributions of this thesis in the area of SD-RAN. It also elaborates the contents of the performed studies and emphasizes the main findings. The overview of the structure of this thesis is portrayed in Fig. 1.1. Research covers four parts mainly: **R1)** definition, architecture design and optimization algorithms for RAN slicing, **R2)** measurements with SD-RAN controller platforms, analysis of SD-RAN control plane design and performance evaluation of the user QoS, and **R3)** proposals of the system architecture for enabling slicing in multi-network environments such as cellular, satellite and aeronautical. All above mentioned research challenges originate from the main challenge that is **R4)** the integration of SD-RAN into the existing 5G network infrastructure.

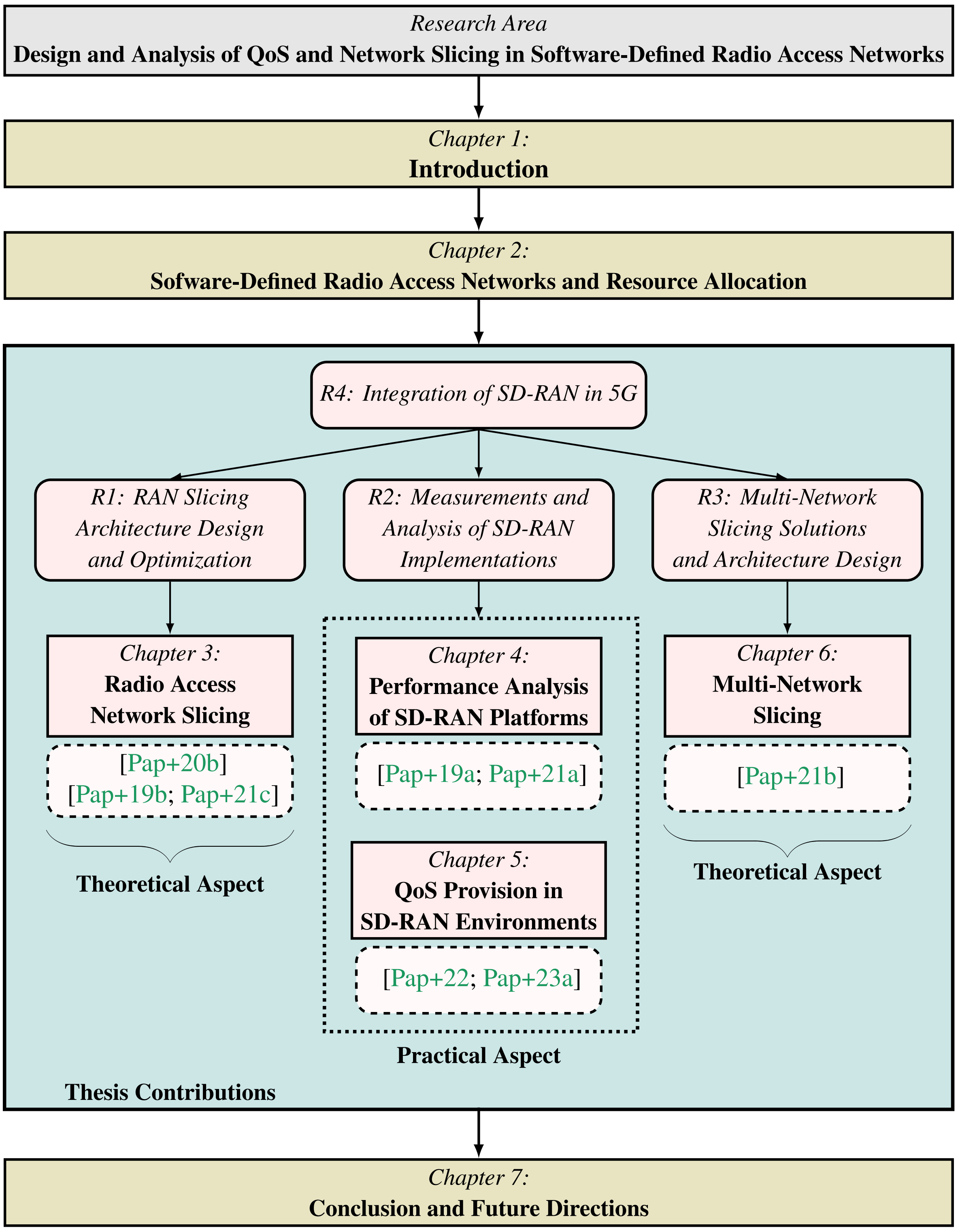

Research Area
Design and Analysis of QoS and Network Slicing in Software-Defined Radio Access Networks
Chapter 1:
Introduction
Chapter 2:
Sofware-Defined Radio Access Networks and Resource Allocation
R4: Integration of SD-RAN in 5G
R1: RAN Slicing Architecture Design and Optimization
R2: Measurements and Analysis of SD-RAN Implementations
R3: Multi-Network Slicing Solutions and Architecture Design
Chapter 3:
Radio Access Network Slicing
[Pap+20b]
[Pap+19b; Pap+21c]
Theoretical Aspect
Chapter 4:
Performance Analysis of SD-RAN Platforms
[Pap+19a; Pap+21a]
Chapter 5:
QoS Provision in SD-RAN Environments
[Pap+22; Pap+23a]
Practical Aspect
Chapter 6:
Multi-Network Slicing
[Pap+21b]
Theoretical Aspect
Thesis Contributions
Chapter 7:
Conclusion and Future Directions

Figure 1.1: Thesis structure. The thesis is mainly addressing four research challenges: **R1)** architecture design and optimization solutions for RAN slicing, **R2)** performance analysis and QoS evaluation of SD-RAN environments, **R3)** slicing concepts for multi-network environments as well as **R4)** SD-RAN integration into the 5G ecosystem and architecture. While the **R1**, **R3** and **R4** are tackled providing theoretical concepts, **R2** focuses on practical implementations and proof-of-concept platforms to verify the developed techniques in a real-based system.

On the one hand, 5G RAN slicing envisions a solution that flexibly deploys heterogeneous services as different **network slices** that can share the same infrastructure. On the other hand, this level of flexibility renders slice management complex, mainly due to the stochastic nature of wireless channels and coupling among network slices. As a solution, SD-RAN is proposed to foster management and orchestration of network slices. Yet, the decoupling of the control and data plane of RAN requires new architecture design concepts to allow for efficient network slice resource allocation and provide strong slice isolation, which is currently limited. Consequently, we initially propose a framework to tackle the above mentioned issue in [Pap+19b]. The proposed method envisions an architecture design where SD-RAN controllers collaborate with **slice managers** (i.e., entities responsible for a slice within a BS) in a joint fashion to enable a dynamic and adaptive wireless resource allocation depending on network slice requirements. The solution utilizes a stochastic optimization method, known as Lyapunov optimization [Nee10], which leads to the satisfaction of network slice requirements while considering a downlink scenario of a single-BS system.

While considering a multi-BS environment, problems related to wireless resource interference, mainly originating due to the overlap of allocated resources, occur. In RAN slicing, the interference management is even more complex, specifically due to the lack of coordination and exchange of information among different network slices operated individually by VMNOs, which are unaware of each other. In order to tackle the interference management in a multi-BS scenario, we design and propose an architecture in [Pap+21c], whereas we extend the Lyapunov optimization approach introduced in [Pap+19b] to cater for interference among BSs managed by different VMNOs. In [Pap+21c], a clear network slice isolation definition is an additional contribution. Up-to-date slice isolation in the literature was related to the ability of meeting slice requirements. However, instead of considering performance guarantees on per-user basis, in [CNS18; SDC19; DOr+19] mostly the overall slice performance guarantee is the focus. Hence, users within a slice might not receive the adequate amount of required resources. Our proposal [Pap+21c] takes this effect into account and provides performance guarantees on lower granularity level (i.e., on user basis).

The second major contribution of this thesis lies in benchmarking of existing SD-RAN controller platforms and evaluation of control plane design characteristics on the user QoS performance. To achieve this, we initially propose in [Pap+19a] a benchmarking tool that elaborates performance insights for one of the most important SD-RAN controller platforms in the literature [Fou+16]. Our approach [Pap+19a] evaluates Central Processing Unit (CPU) and memory consumption as well as it provides delay measurements for [Fou+16], while considering varying number of BSs in the data plane. To extend the performance evaluation towards a more representative range of SD-RAN controller platforms, in [Pap+21a] we propose `MARC`, a novel benchmarking tool for SD-RAN architectures and their controllers.

We use MARC to measure, analyze and identify performance implications for two state-of-the-art open-source SD-RAN solutions [Fou+16; CKR19]. We perceive results for monitoring application scenarios, considering fully centralized control, which demonstrates that the proposed architectures with a single SD-RAN controller are not scalable and they can even lead to unpredictable network operations.

To overcome the issue of centralized control approaches, a distributed SD-RAN control plane is necessary. However, a distributed control plane renders the management in the control plane complex and incurs additional overhead, for instance control handovers. In order to investigate the effect of SD-RAN control plane design on the user performance, extensive simulation-based results are provided to demonstrate the benefits and drawbacks of the two approaches in [Pap+22]. Finally, to evaluate the impact on a real-based system, in [Pap+23a] measurements are performed on a real SD-RAN controller [Mosa]. In [Pap+23a] additionally, a theoretical method is proposed to capture the effect of control plane inconsistencies on the users in the data plane of SD-RAN environments.

The final contribution of this thesis lies on the analysis, design and optimization of network slicing concepts for multi-network environments such as cellular, satellite and aeronautical, in the support of In-Flight Entertainment and Connectivity Services (IFECS). In that regard, in [Pap+21b] we propose a novel system for providing network slicing in aircrafts based on SD-RAN features. We further introduce a new definition for network slicing based on the willingness of Service Provider (SP)s to reveal information about their costumers. Thus, we provide results with respect to the cost of network slicing.

1.3 Outline

The remainder of the thesis is structured as follows (in line with Fig. 1.1).

Chapter 2 provides a general overview on the SD-RAN concept and architecture description. Furthermore, it gives background on the 5G RAN infrastructure as well as RAN resource allocation. Finally, it introduces a summary of network slicing and describes the relation of SD-RAN and network slicing into the context of the 5G RAN ecosystem.

Chapter 3, as shown in Fig. 1.1, deals with the research question **R1**. It addresses the concept of RAN slicing in depth. First, it provides a clear definition of slice isolation and differentiation from traditional MAC scheduling. Additionally, a system design is presented to encapsulate the SD-RAN and network slicing concepts into the 5G ecosystem in the context of wireless resource allocation. Novel algorithms are further introduced to provide RAN slicing in a single-BS scenario with distinct network slice requirements and dynamic wireless channel variations. The optimization aims at wireless resource minimization while satisfying slice requirements and providing slice isolation. Moreover, an extended approach towards a multi-BS scenario is presented, where a comparison with other literature solutions, like [DOr+19] is provided with respect to the achieved overall slice and individual user throughput.

Chapter 4 investigates the performance predictability of SD-RAN controller platforms such as [Fou+16; CKR19]. It gives insights into the performance bottlenecks of the considered controllers through a novel SD-RAN benchmarking tool, namely `MARC`. Furthermore, the design and implementation details of `MARC` are elaborated. Finally, a summary of the SD-RAN controllers' capabilities is presented.

Chapter 5 initially proposes a proof-of-concept, namely `Delphi`. `Delphi` provides predictable throughput performance evaluations for users in SD-RAN environments under varying network conditions. This is achieved with mathematical models, simulations and measurements. Moreover, Chapter 5 studies the effect of SD-RAN control plane design on the user QoS. It initially provides an architecture design for a distributed SD-RAN control plane, including control handovers. Further, it discusses a novel SD-RAN simulation platform based on 5G system features. Using the designed simulator, extensive evaluations are performed to highlight the user performance and compare the centralized and distributed SD-RAN control plane architectures. Results presented in both Chapter 4 and Chapter 5 address the research question **R2**, as explained in Section 1.1.

Chapter 6 deals with research question **R3**, as shown in Fig. 1.1. It is mainly concerned with a system design and development of optimization approaches for the provisioning of the network slicing concept on multi-network environments. It further introduces a novel slice concept based on the willingness of SPs to provide user details. The solutions for different slice requirements are obtained by solving a Mixed-Integer Linear Programming (MILP).

The research questions tackled in Chapters 3 to 6 originate mainly from **R4**, as demonstrated in Fig. 1.1.

Chapter 7 concludes the thesis, where an overview is presented and potential future work is discussed.

Chapter 2

Towards Programmable and Softwarized 5G Networks

In this chapter, we elaborate on the general technologies and concepts related to this thesis. Since the main goal of this thesis is to improve QoS in 5G networks, initially, details with respect to the 5G network architecture are presented in Section 2.1. This is done by describing in depth the 5GC and NR functionalities and highlighting the main differences with LTE. Since SDN, and in particular SD-RAN, are considered to be the pillars in the context of provisioning the required level of flexibility in 5G networks, Section 2.2 introduces the main concepts of SDN and SD-RAN while outlining their differences. Given the importance of network slicing in the core network and RAN with respect to the ability of achieving the required level of network utilization, Section 2.3 presents these concepts. As the main focus of this thesis is on RAN, particular emphasis is put on RAN slicing. Finally, Section 2.4 provides a summary of this chapter.

2.1 An Overview of 5G

In this section, we first present the 5G System Architecture (5GS) architecture. Moreover, we elaborate on the 5G functions for both the core network and RAN.

2.1.1 5G System Architecture

The high-level overview of the 5GS is depicted in Fig. 2.1. The architecture is split into two main parts, namely the 5GC, where the 5G core functionalities lie and the NR, where the 5G RAN components are located [ETS].

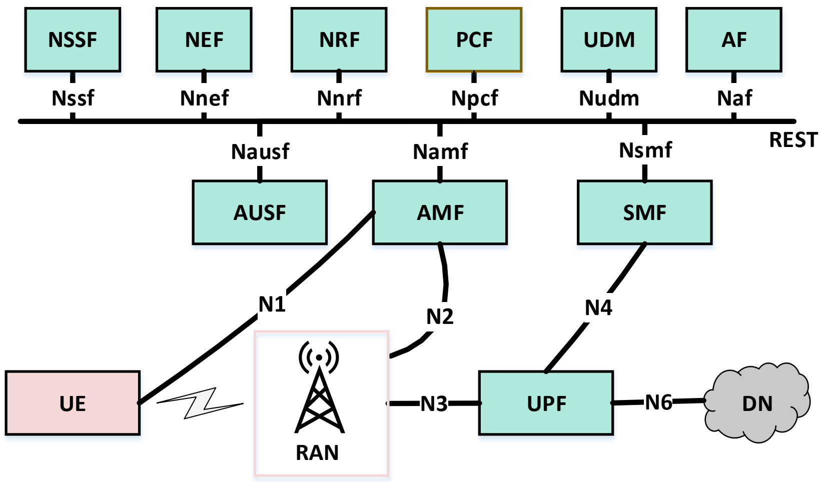


Figure 2.1: Overview of the 5GS concept, where the 5GC functions are depicted with cyan color, whereas the 5G NR functions are depicted in rose. Furthermore, interfaces among functionalities are presented in black. Finally, the Data Network (DN) is portrayed in gray color.

2.1.2 5G Core Network Functions

While the 5GC builds upon its 4G counterpart architecture concept, it still introduces two main novelties. The first novelty consists on a clear separation between control plane and data plane, whereas the second lies in the provisioning of the functions in terms of a service-based architecture [3GPa]. In that regard, the main components of the 4G core, such as Mobility Management Entity (MME), Service Gateway (SGW), Packet Gateway (PGW), Home Subscriber Server (HSS) and Policy Charging Rules Function (PCRF) are kept in the 5GC. Yet, their functionalities are distributed across multiple components. Furthermore, to elaborate new emerging technologies, the 5GS envisions functions to enable and manage network slicing. The principal 5GC functions are detailed as follows:

1. **Authentication Mobility Function (AMF):** responsible for the management, establishment and the maintenance of the communication to the UE, as well as security and mobility.

2. **User Plane Function (UPF):** responsible for mobility anchoring and packet handling. It can further facilitate the packet forwarding, routing and inspection.

3. **Session Management Function (SMF):** responsible for the UE's Internet Protocol (IP) address allocation and selection of the appropriate UPF for each network slice.

4. **Unified Data Management (UDM):** database that stores all the information concerning the subscribers of the network.

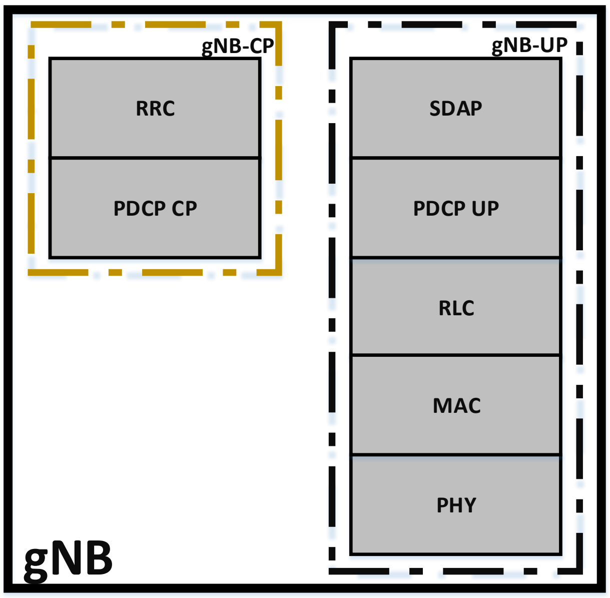


Figure 2.2: Overview of NR functions mainly realized in the gNB. We distinguish between control plane (gNB-Control Plane (CP)) and user plane (gNB-User Plane (UP)) functions.

5. **Authentication Server Function (AUSF):** carries user authentication functionalities.
6. **Policy Control Function (PCF):** responsible for applying network policies.
7. **Network Exposure Function (NEF):** responsible for exposing the capabilities of the network functions to candidate applications or third parties. The range of exposure can consist of monitoring, provisioning and traffic routing.
8. **Network Slicing Selection Function (NSSF):** facilitates the selection of the network slices.
9. **Network Repository Function (NRF):** responsible for advertising NFs with management tasks such as registration, de-registration, authentication, discovery.
10. **Application Function (AF):** responsible for handling various applications that can be offered to the network.

2.1.3 5G Radio Access Network

Here, as part of the 5G RAN description, initially the NR protocol stack is presented, followed by an elaboration on the NR resource grid structure.

2.1.3.1 NR Protocol Stack

The overview of the NR protocol stack is depicted in Fig. 2.2. The main entity of the NR is referred to as the gNB (i.e., BS in 5G). Within the gNB, there exist several functionalities [3GP18b]. These functionalities are further divided into control plane and user/data plane, which we explain in the following:

1. **Radio Resource Control (RRC):** responsible for handling RAN control plane functionalities such as connection setup/release, control steering to 5GC (i.e., AMF), AMF selection upon UE arrival, mobility control, broadcasting of system information to UE (i.e., Signal Radio Bearer (SRB)s), radio admission control.
2. **Service Data Application Protocol (SDAP):** concerns the conversion of QoS flows to Data Radio Bearer (DRB)s.
3. **Packet Data Convergence Protocol (PDCP):** responsible for tasks such as IP header compression, retransmissions and packet duplication handling during handovers.
4. **Radio Link Control (RLC):** in charge of tasks that deal with packet segmentations.
5. **MAC:** its responsibilities include scheduling and logical channel multiplexing.
6. **Physical Layer (PHY):** it deals with coding/decoding and modulation/demodulation.

2.1.3.2 NR Resource Grid

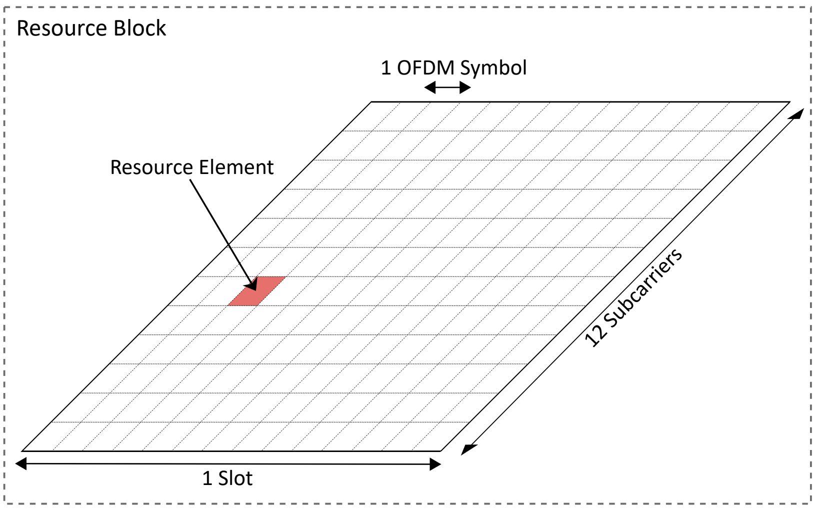


Figure 2.3: Overview of a Physical Resource Block (PRB) in 5G NR. A PRB consists of 12 subcarriers in the frequency domain and 1 slot in the time domain.

Similarly to LTE, the 5G RAN counterpart, which is referred to as NR, maintains Orthogonal Frequency Division Multiplexing (OFDM) as the multiplexing technology both for the downlink and uplink. The OFDM signal consists of multiple subcarriers spaced from each other by 15 kHz, known as the *baseline* approach for NR. Yet, given the focus of 5G in flexibility, subcarrier spacing is tunable and can vary generally from 15 kHz to 240 kHz. In the time domain, transmissions occur on a frame basis consisting of 10 ms. In turn, every frame is further divided into 10 subframes of 1 ms each. Within a subframe several slots may exist depending on the subcarrier spacing configuration. Each slot contains 14 OFDM symbols, whose duration reduces with increasing subcarrier spacing. As a result, compared to LTE, where a slot duration has been fixed to 1 ms, lower slot duration implies faster transmission and therefore fosters URLLC applications.

The minimum resource allocation granularity in NR is referred to as PRB. One PRB consists of 12 subcarriers in the frequency domain and 1 slot in the time domain. Within a slot, a PRB contains 14 OFDM symbols, as shown in Fig. 2.3. The smallest component inside the PRB is known as Resource Element (RE) and consists of 1 OFDM symbol and 1 subcarrier. Given a specific subcarrier spacing, the frequency and time allocation of a PRB varies. For instance, for a subcarrier spacing of 15 kHz, a PRB is assigned 180 kHz in the frequency and 1 ms in the time domain, respectively. Alternatively, a subcarrier spacing of 30 kHz, denotes that a PRB contains 360 kHz in the frequency and 0.5 ms in the time domain. Finally, the total amount of PRBs in a system directly depends on the total available transmission bandwidth. Whereas in LTE the maximum available carrier bandwidth was 20 MHz, 5G envisions up to 400 MHz for one carrier. More detailed description can be found in [Dah18].

2.2 Software-Defined Networking in 5G

This section introduces the concept of SDN in 5G. Moreover, it highlights the differences between SDN in the core network as well as in RAN.

2.2.1 Software-Defined Networking

With the promise of network programmability and flexibility, SDN has been a paradigm shift that has led research in the last couple of years [Jar+14; Kre+14; Kel+19].

Whereas traditional networking devices such as switches and routers contained the data plane and control plane within a single component, SDN envisions a separation among the two. To that end, the control is centralized in entities referred to as SDN controllers [Ryu; Med+14; Ber+14a], which reside in the control plane. The SDN controller maintains a broader network view and is in charge of all the control decisions. An overview of the SDN architecture is illustrated in Fig. 2.4. The control plane containing the SDN controllers can be either *centralized* or *distributed*. While the centralized control plane introduces less overhead, it is often prone to single point of failures [KD17]. Therefore, approaches to overcome these issues, which increase the network scalability are proposed [Med+14].

According to Fig. 2.4, traditional network components such as switches reside in the data plane. Since they do not possess any control capabilities, they only obey to the rules imposed by the SDN controller. For the communication among the SDN controller in the control plane and the devices in the data plane, a Southbound Interface (SBI) such as [McK+08] is utilized. Additionally, the SDN controller also communicates with applications in the **application plane** through a Northbound Interface (NBI), for instance Representational State Transfer (REST) Application Programmable Interface (API) [Sez+13]. The applications can range from **deep packet inspection** to **firewalls** and **load balancers**.

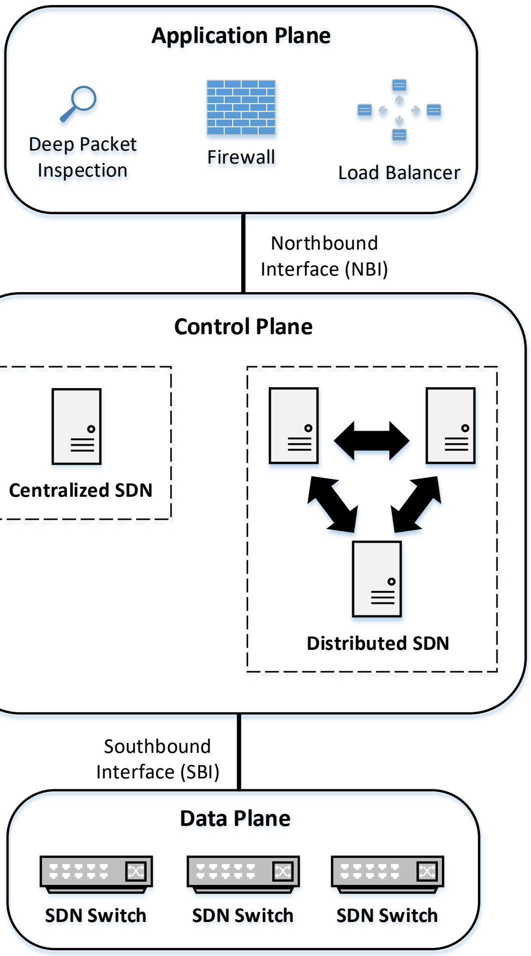


Figure 2.4: Overview of the SDN concept. SDN controllers in the control plane enforce rules on switches in the data plane based on applications. The SBI and NBI are utilized for the communication between the control plane and data plane, as well as control plane and application plane, respectively.

2.2.2 Software-Defined Radio Access Networks

Similar to the concept of SDN in the core network, softwarization in RAN referred to as SD-RAN, implies control functionality over the RAN. This has triggered a vast ongoing research, where SD-RAN prototypes from academia [Fou+16; CKR19; FMK17; SIN21], but also industry such as the O-RAN alliance [O-Ra] have been introduced.

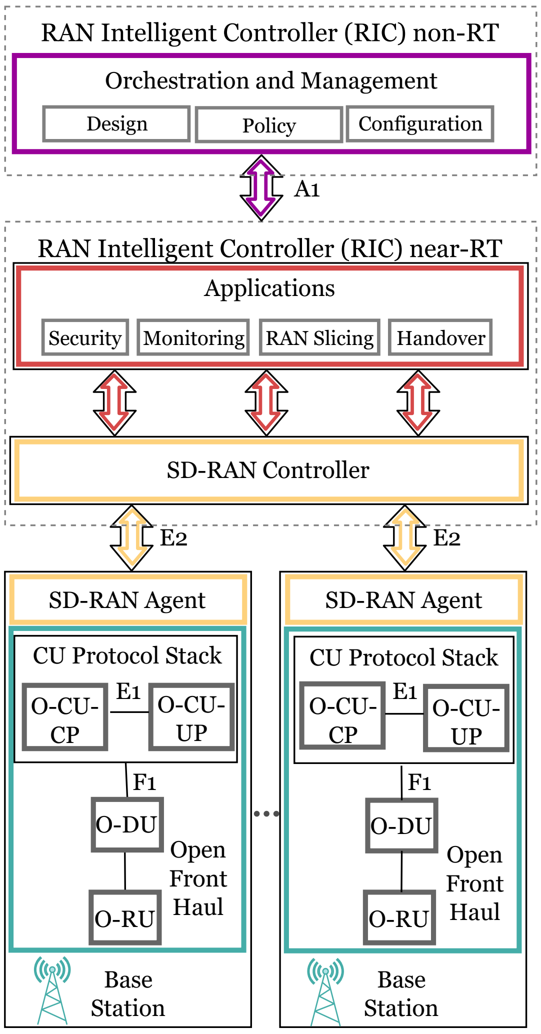


Figure 2.5: Overview of the SD-RAN concept, where SD-RAN controllers in the control plane imply rules on RAN BSs in the data plane with respect to applications such as MAC scheduling.

The concept of SD-RAN introduces the separation of the control plane and user/data plane of RAN to foster programmability and softwarization. As illustrated in Fig. 2.2, there already exists a separation of gNB functionalities between control plane and user plane, where functions such as RRC lie in the control plane. Additionally, the SD-RAN concept introduces a further separation in the control plane logic, where decisions for specific control functionalities such as *handover*, *interference management*, and *scheduling* are taken by devices referred to as SD-RAN controllers. In turn, their decisions are enforced in the gNB by control plane functions such as RRC.

A high level overview of the SD-RAN paradigm is given in Fig. 2.5. As presented, SD-RAN controllers abstract the underlying network for the applications. The nature of such applications can vary from monitoring to operational and management. To facilitate the communication between the SD-RAN controllers and the underlying network, BSs are equipped with control components referred to as SD-RAN agents. Moreover, protocol functions within BSs are structured according to the proposed O-RAN splits [O-Ra], namely, Central Unit Control Plane (CU-CP), Central Unit User Plane (CU-UP), Distributed Unit (DU) and Radio Unit (RU) as demonstrated in Fig. 2.5. However, for the sake of simplicity, for the remainder of the thesis we will refer to a BS as a single component. Moreover, the terms user plane and data plane will be used interchangeably throughout this thesis.

SD-RAN agents follow the SD-RAN protocol and are responsible for translating and enforcing the SD-RAN controllers' decisions on the underlying BS devices. While SD-RAN agents reside at the BSs, SD-RAN controllers can be deployed on edge servers and communicate via Transport Control Protocol (TCP) with the SD-RAN agents. A potential interface, known as E2, is introduced by O-RAN [O-Rd]. Finally, UEs are served by BSs in order to enable communication. The block composed of the SD-RAN controller and applications, is referred to in the O-RAN terminology [O-Ra] as **Near-Real Time RAN Intelligent Controller (near-RT RIC)**. Furthermore, the near-RT RIC [O-Re] is connected with the orchestration and automation layer, referred to as non-Real Time RAN Intelligent Controller (non-RT RIC) [O-Rc]. The interface for this communication, designed by O-RAN, is known as A1 [O-Rb].

2.3 Network Slicing in 5G

This section introduces the concepts of 5GC and RAN slicing and highlights the main definitions in the state of the art.

2.3.1 Core Network Slicing

Network slicing in the core network has been a well investigated topic in the last couple of years. In principle, the network slicing concept in the core network side is tackled by differentiating IP packets of various services and mapping them to appropriate QoS flows. In 4G, a 3GPP standardization by the name Enhanced Dedicated Core (eDECOR) already exists [3GPb].

In the core network side, network slicing solutions are often combined with the SDN paradigm as demonstrated in [Afo+18]. To achieve the adequate QoS requirement, efficient rule updates from the SDN controllers sent towards the network switches are needed [Ble+16]. Moreover, solutions elaborate on optimal allocation approaches for core network resources (i.e., link bandwidth, number of switches, memory and CPU allocation) to achieve efficient slicing [Che+19; Mar+18; Tal+17].

Recently, considering the new architecture model of 5G, as explained in Section 2.1.2, solutions for core network slicing anticipate the optimization of network function placement and efficient network function allocation [SM19; WL21; Zha19].

2.3.2 Radio Access Network Slicing

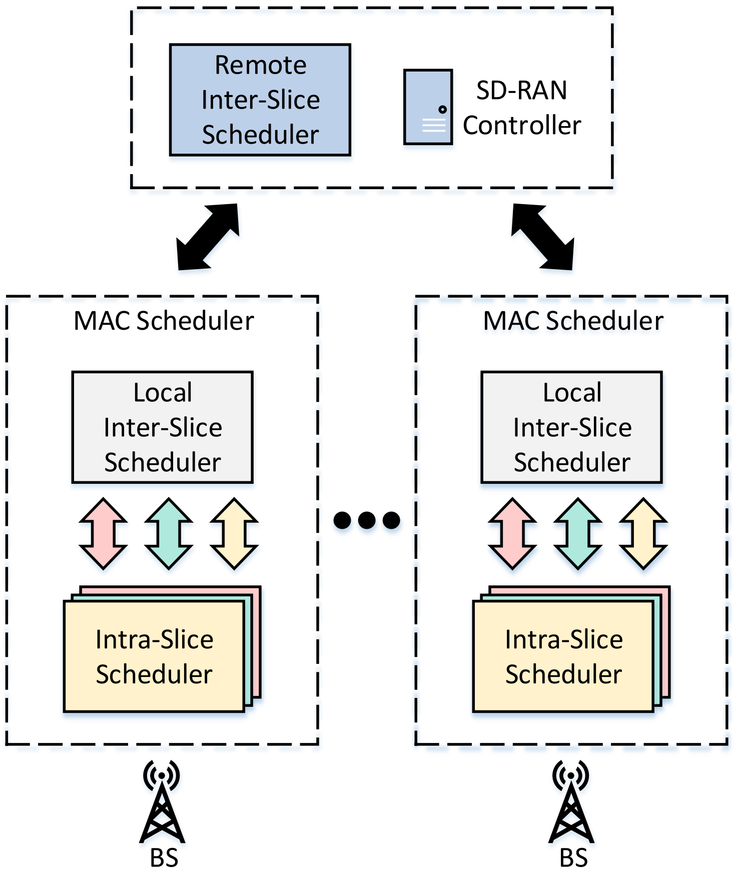


Figure 2.6: Overview of the RAN slicing concept performed in a hierarchical fashion. The first level consists of an inter-slice scheduler, running locally at the BS or remotely at the SD-RAN controller. The second level consists of the intra-slice scheduler.

From an architectural perspective, RAN slicing is associated with the concept of RAN resource sharing among multiple VMNOs. Conceptual works have been proposed to enable this procedure [SCS16; KL14; Sal+17].

The envisioned concept of RAN slicing relies on a hierarchical approach, which is performed in two levels [Kok+11; CNS18; DOr+19; Pap+19b; Pap+21c], as illustrated in Fig. 2.6. The first level consists of what is called the **inter-slice scheduler**, which is responsible for allocating PRBs among RAN slices. In turn, the second level in the hierarchy, namely the **intra-slice scheduler** or **slice manager** is responsible for allocating the received PRBs among the UEs of the **network slice**. While the **intra-slice scheduler** is always located at the BS, the **inter-slice scheduler** can be either locally operated within the BS or remotely at the SD-RAN controller. When operated remotely at the SD-RAN controller, a broader view of the network is obtained and therefore it may lead to enhanced network management and coordination. In this context, for this thesis we adopt the architecture where the inter-slice scheduler is remotely running at the SD-RAN controller.

2.3.2.1 Inter-Slice Scheduler

As already mentioned, the inter-slice scheduler allocates PRBs among network slices. In turn, the inter-slice scheduler can be classified as *local* or *remote*. The former indicates that the inter-slice scheduler is part of the BS and is solely maintaining the slicing resource allocation decision. The latter, implies that the inter-slice scheduler resides in the SD-RAN controller and is therefore responsible for multiple BSs. This can for instance increase the level of coordination among BSs, leading to improved management and coordination [Fou+16; Pap+20b].

The simplest solution for inter-slice scheduler assumes a *static* approach. In this scenario, PRBs have a fixed mapping into network slices, which does not change over time. While this solution provides strong isolation among network slices, it does not allow for multiplexing among them. This can be beneficial in terms of resource utilization, as network resources that are not needed by a network slice can be shared among other network slices. Thus, the introduction of an alternative method, which considers a more *dynamic* approach becomes necessary. That implies that PRB allocation among network slices changes over time depending on network slice requirements and network conditions.

2.3.2.2 Intra-Slice Scheduler

Whereas the inter-slice scheduler deals with PRB allocation among network slices, the intra-slice scheduler assigns the required PRBs from each network slice to its respective UEs. The problem of intra-slice scheduling is traditionally known as MAC scheduling and has been well investigated in the literature.

Typical intra-slice schedulers range from *round robin*, where PRBs are allocated to UEs one by one in a circular fashion to *maximum throughput*, where PRBs are assigned to those UEs that achieve the highest throughput. More details on intra-slice schedulers from the mathematical perspective can be found in [FL02; Cap+12], whereas for details from the implementation point of view we refer the readers to [UPK22; Nik+14; Gom+16].

2.4 Summary

In this chapter, initially, background information on the architecture design of 5G networks was discussed. To that end, a detailed description and an overview of the 5GC and NR was provided. Since the main focus of this thesis lies on RAN, particular emphasis is put on the concepts of NR protocol stack and resource grid, which are crucial for the resource allocation problem in RAN. Furthermore, this chapter also introduces the concepts of SDN in core network and SD-RAN, which will be the basis of the enhancements provided in 5G with respect to programmability and softwarization. Finally, a detailed description of RAN slicing, which is one of the main contributions of this thesis, is demystified. Given the obtained background, the following chapters of this thesis introduce architectural designs, mathematical models and optimization solutions to enhance the capabilities of 5G networks and improve the QoS of users in the network.

Chapter 3

Definition, Design and Optimization of Radio Access Network Slicing

While SDN can ease the resource management and orchestration of 5G networks, it still cannot accommodate standalone the heterogeneity of emerging applications. To address the above issue, the concept of RAN slicing has emerged. RAN slicing enables the coexistence of multiple VMNOs or third parties such as vehicular industries, factories and aircrafts that share the same physical infrastructure. In such a setup, each slice should remain unaffected by other slices, preserving isolation [All16].

On the one hand, RAN slicing enables the coexistence of multiple VMNOs or third parties as different *slices* of the network. On the other hand, coexistence brings the requirement of isolation among the created network slices, which presents the decoupling of slices from each others' state of operation. In that context, one of the main challenges of RAN slicing, the ability to operate in an isolated manner and stay unaffected by other slices, is a crucial, but a non-trivial task [Fou+17; Zha+17]. Furthermore, additional RAN slicing challenges originate mainly from the stochastic nature of wireless channels, and the tight resource coupling among slices. Therefore, there is a need to consider optimization techniques that capture all the intricacies of wireless channels. Obviously, the static solutions that assume a unique and unchanged resource allocation do not suffice due to the constantly changing nature of wireless channels. Consequently, in this chapter we develop and analyze in detail different methods that envision and enable dynamic reconfigurations of wireless resource allocation.

From the standardization perspective, RAN slicing in combination with the SD-RAN concept still remains an interesting, but not fully mature topic compared to traditional SDN and core network slicing approaches for which a 3GPP standardization already exists [3GPb]. In that regard, in this chapter, based on our theoretical models and system design, we provide a proposal for the integration of the SD-RAN concept with respect to RAN slicing as part of the 3GPP standardization.

Content and outline of this chapter: Section 3.1 provides the system model used for RAN slicing and elaborates the details of the simulator and parameters used for the performance evaluation. Furthermore, it introduces the SD-RAN-based RAN slicing approach, discussing all system components and their interaction to enable RAN slicing. Finally, it reveals related work on RAN slicing both for single and multi-BS scenarios.

Initially in Section 3.2, we focus on a case study for a single-BS scenario with respect to RAN slicing aiming at minimizing resource usage while satisfying slicing constraints. Considering total system throughput maximization while maintaining absolute slice isolation on the UE level, we portray a problem formulation in Section 3.3. In this section, we provide a clear definition of slice isolation and take advantage of wireless channel characteristics to dynamically re-assign wireless resources to network slices. In this way, we are able to outperform those state-of-the-art solutions that do not take into account the stochastic nature of wireless environments. Besides proposing theoretical models, Section 3.4 presents an initial concept for a practical implementation of the proposed solutions. Finally, Section 3.5, combines the above mentioned approaches and envisions a RAN control function based on 3GPP standardization for the provisioning of RAN slicing.

The content of this chapter is based on the following publications. The results of the single-BS scenario with aim at resource usage minimization is presented in [Pap+19b]. Moreover, the definition of slice isolation and the extension towards multiple BSs achieving maximum system throughput while maintaining a minimum user throughput requirement is demonstrated in [Pap+21c]. The concept of integrating the above solutions as a additional RAN function in the 5G architecture is shown in [Pap+20b].

[Pap+19b] A. Papa, M. Klugel, L. Goratti, T. Rasheed, and W. Kellerer. "Optimizing dynamic RAN slicing in programmable 5G networks." In: Proc. of the IEEE International Conference on Communications (ICC). 2019, pp. 1–7. doi: 10.1109/ICC.2019.8761163.

[Pap+21c] A. Papa, A. Jano, S. Ayvaşık, O. Ayan, H. M. Gürsu, and W. Kellerer. "User-Based Quality of Service Aware Multi-Cell Radio Access Network Slicing." In: IEEE Transactions on Network and Service Management (2021). doi: 10.1109/TNSM.2021.3122230.

[Pap+20b] A. Papa, R. Durner, L. Goratti, T. Rasheed, and W. Kellerer. "Controlling Next-Generation Software-Defined RANs." In: IEEE Communications Magazine (2020), pp. 58–64. doi: 10.1109/MCOM.001.1900732.

3.1 System Model and Related Work

To ease the understanding and readability of this thesis, this section reveals background on the system model and simulation environment used for the performance evaluation. Furthermore, it introduces the SD-RAN-based RAN slicing architecture and the interaction among system components. Additionally, related work focusing on RAN slicing in single and multi-BS environments is discussed, highlighting the differences of our methods compared with the state of the art. Before proceeding further, Table 3.1 summarizes the notation of this chapter.

3.1.1 System Model

In this thesis, we focus on a downlink scenario of a cellular network consisting of a set $\mathcal{B}$ of BSs. Following the realistic assumption of limited spectrum bands for commercial use [Fed], in our scenario adjacent BSs are interfering with each other (i.e., frequency reuse factor of 1). The RAN system belongs to a single Infrastructure Provider (InP), but can be leased to other VMNOs or third parties (i.e., aircrafts, automotive industries, health care providers) referred to as **network slice owners** or **tenants**. We assume that in our system we have a set $\mathcal{S}$ of S network slices. To enable flexibility, slices can be deployed across multiple BSs. The RAN serves a total number of N UEs from a set $\mathcal{N}$. We define δ_i^s as a binary variable being 1 if UE i belongs to slice s and 0 otherwise. The slice owner knows the value of this parameter from the beginning. Each UE in the network can be attached to only one network slice s, i.e., $\sum_s \delta_i^s = 1, \forall i \in \mathcal{N}$ and it can only be served by one BS $b \in \mathcal{B}$ at a time. We consider a time slotted system of T slots from a set $\mathcal{T}$, where each slot $t \in \mathcal{T}$ corresponds to the Transmission Time Interval (TTI), which is 1 ms in LTE as well as for 5G considering a subcarrier spacing of 15 kHz, (see Section 2.1.3.2). The spectrum is divided into a set $\mathcal{R}$ of R PRBs.

Let $h_{i,j}^b(t)$ represent the channel gain of UE $i \in \mathcal{N}$, on PRB $j \in \mathcal{R}$ for BS $b \in \mathcal{B}$ in time slot $t \in \mathcal{T}$. Moreover, we assume that the channel gain remains constant for the coherence time of the channel τ and changes every multiple of τ for all the UEs in the system. Throughout this chapter, τ corresponds to 20 ms, as reported in [MC08]. Let variable $w_{i,j}^b(t)$ be a decision binary variable taking the value 1 if UE $i \in \mathcal{N}$ obtains PRB $j \in \mathcal{R}$ from BS $b \in \mathcal{B}$ in time slot $t \in \mathcal{T}$, or 0 otherwise. We then define $w_{i,j}^{b,s}(t) = w_{i,j}^b(t) \cdot \delta_i^s$ as a variable that captures the outcome of the resource allocation per slice. In other words, for the PRB $j \in \mathcal{R}$, part of slice $s \in \mathcal{S}$, to be allocated to UE $i \in \mathcal{N}$, the latter has to belong to that slice.

Let the variable $p_j^b(t)$ denote the power allocation for the transmission on PRB $j \in \mathcal{R}$ of BS $b \in \mathcal{B}$ in slot $t \in \mathcal{T}$. For the sake of simplicity, we assume the transmission power to be shared equally among all PRBs in a BS $b \in \mathcal{N}$. Further, we denote the variable N_0 as the thermal noise, whereas W as the bandwidth of each PRB i.e., 180 kHz in our scenario. Finally,

Table 3.1: Notation of system parameters and variables

Variable	Description
$\mathcal{N}$	Set of network users
$\mathcal{S}$	Set of network slices
δ_i^s	Variable that indicates whether user $i \in \mathcal{N}$ belongs to slice $s \in \mathcal{S}$
$\mathcal{B}$	Set of BSs in the system
η_s^b	Variable that indicates whether slice $s \in \mathcal{S}$ can be served by BS $b \in \mathcal{B}$ or not
$\mathcal{T}$	Set of slots in the system
$\mathcal{R}$	Set of PRBs in the system
$h_{i,j}^b(t)$	Wireless channel gain for user $i \in \mathcal{N}$ at PRB $j \in \mathcal{R}$ for BS $b \in \mathcal{B}$ in slot $t \in \mathcal{T}$
τ	Wireless channel coherence time
$w_{i,j}^b(t)$	Variable that indicates if user $i \in \mathcal{N}$ served by BS $b \in \mathcal{B}$ at slot $t \in \mathcal{T}$ receives PRB $j \in \mathcal{R}$ or not
$w_{i,j}^{b,s}(t)$	Variable that indicates if user $i \in \mathcal{N}$ of slice $s \in \mathcal{S}$ served by BS $b \in \mathcal{B}$ at slot $t \in \mathcal{T}$ receives PRB $j \in \mathcal{R}$ or not
$p_j(t)$	Power allocation to PRB $j \in \mathcal{R}$ in slot $t \in \mathcal{T}$
I	Interference
W	Bandwidth of a PRB
N_O	Thermal noise
$r_{i,j}^b(t)$	Data rate of user $i \in \mathcal{N}$ served by BS $b \in \mathcal{B}$ in slot $t \in \mathcal{T}$, given PRB $j \in \mathcal{R}$
$r_i(t)$	Overall data rate of user $i \in \mathcal{N}$ in slot $t \in \mathcal{T}$
$r_s(t)$	Overall data rate of slice $s \in \mathcal{S}$ in slot $t \in \mathcal{T}$
D_s	Average delay of slice $s \in \mathcal{S}$
$K_s(t)$	Total number of PRBs assigned to slice $s \in \mathcal{S}$ in slot $t \in \mathcal{T}$
ρ_s	Ratio of resources requested by slice $s \in \mathcal{S}$
$\underline{\gamma}_i^s$	Rate policy for user $i \in \mathcal{N}$ within slice $s \in \mathcal{S}$
$\overline{C}_s$	Maximum allowed bandwidth for slice $s \in \mathcal{S}$
D_s^{max}	Maximum allowed delay for slice $s \in \mathcal{S}$
$\underline{C}_s$	Minimum required bandwidth for slice $s \in \mathcal{S}$
$U_s(t)$	Backlog queue for the incoming traffic of slice $s \in \mathcal{S}$ in slot $t \in \mathcal{T}$
$\lambda_s(t)$	Arrival rate of the incoming traffic for slice $s \in \mathcal{S}$ in slot $t \in \mathcal{T}$
$\alpha_s(t)$	Admitted traffic for slice $s \in \mathcal{S}$ in slot $t \in \mathcal{T}$
V	Lyapunov design parameter
$\Theta(t)$	System queue state in slot $t \in \mathcal{T}$

we represent the interference experienced by each UE by the variable I, which is composed by the rate of each interfering BS to UE $i \in \mathcal{N}$ served by BS $b \in \mathcal{B}$ and calculated as

$$I = \sum_{\beta \in \mathcal{B} \setminus \{b\}} h_{i,j}^{\beta}(t) \cdot p_j^{\beta}(t). \tag{3.1}$$

In turn, for the data rate that UE $i \in \mathcal{N}$ receives from PRB $j \in \mathcal{R}$ in BS $b \in \mathcal{B}$ we can write

$$r_{i,j}^{b}(t) = W \log_2 \left(1 + \frac{h_{i,j}^{b}(t) \cdot p_{j}^{b}(t)}{I + N_0}\right). \tag{3.2}$$

Consequently, the rate that each UE $i \in \mathcal{N}$ achieves in slot $t \in \mathcal{T}$ can then be expressed as

$$r_i(t) = \sum_{b=1}^{B} \sum_{s=1}^{S} \sum_{j=1}^{R} w_{i,j}^{b,s}(t) \cdot r_{i,j}^{b}(t). \tag{3.3}$$

Finally, we can express the total throughput that each slice achieves as the accumulation of individual slice's UEs throughputs as

$$r_s(t) = \sum_{i=1}^{N} r_i(t). \tag{3.4}$$

In the following, we proceed next with the description of the simulation environment.

Table 3.2: Simulation parameters

Input Parameters	**Value**
System bandwidth W per BS	5 MHz
Carrier Frequency	1.8 GHz
BS antenna horizontal/vertical	70°/10° beam width
BS antenna down-tilt	15°
Number of PRBs per BS	25
Transmission power per BS	−15 dBm
Shadowing	Gaussian zero-mean with 4.8 dB standard deviation (σ_L) denoted by the variable X
F_{d_r} is the free space loss at d_r	37.5 dB
Reference d_r	1 m
Pathloss exponent (n)	2.6
Multipath	Rice distribution with a mean of -1.4 dB and K-factor of 8.1 dB
Path loss in dB at distance d	$PL = F_{d_r} + 10n\log(\frac{d}{d_r}) + X(0, \sigma_L)$
Number of slots	5000

3.1.2 Simulation Environment

Given the importance of small cells for 5G networks [Chr19; Hua16; Nok17] and the challenging scenario they represent (i.e., high interference among BSs), for our evaluations we consider an in-aircraft channel model as a 5G small cell representative use case based on realistic measurements performed within an aircraft cabin [Mor+].

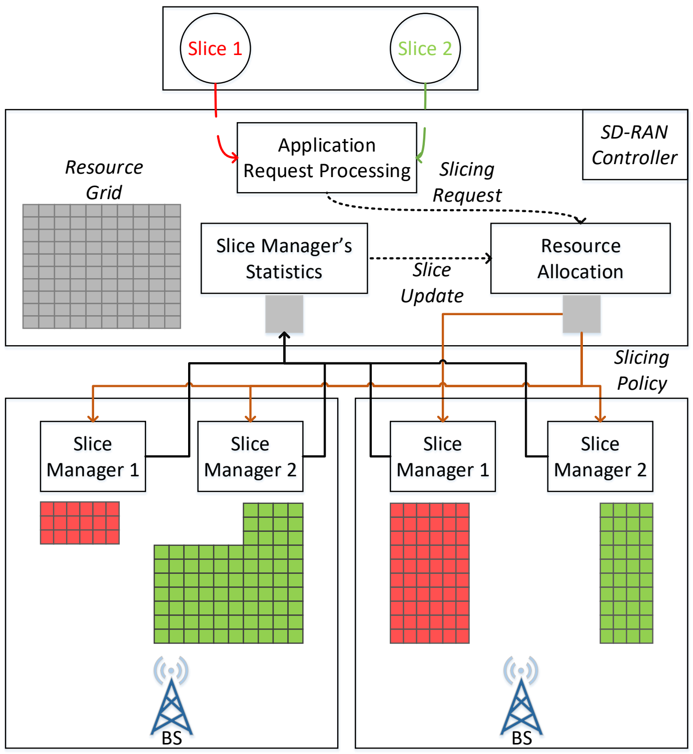


Figure 3.1: Multi-BS SD-RAN-enabled RAN slicing architecture. The SD-RAN controller is the heart of the architecture. For every slice, a slice manager is created that communicates with the SD-RAN controller and provides efficient resource allocation for network slices.

Our simulation environment is Matlab-based, running on a Dell server Intel(R) Xeon(R) E5-2650 CPU, with 12 physical cores at frequency 2.20 GHz and 64 GB of Random Access Memory (RAM). UEs are distributed according to the seat plan of a Boeing B737-400 [Boe], where 156 passengers are placed in 26 rows of 6 seats each. Furthermore, up to 5 BSs are positioned in the middle of the aircraft and they all contain bi-directional antennas. For every UE within the simulation, channel characteristics are generated according to the presented channel model in Table 3.2 based on measurements from [Mor+]. To mimic the dynamic characteristics, the wireless channel conditions i.e., τ, change every 20 ms [MC08]. The simulation contains 5000 slots per each iteration.

Here we stress that although we present our evaluations for an in-aircraft use case, our approach is applicable and beneficial for any multi-BS scenario with time and frequency channel variations.

3.1.3 SDN-enabled RAN Slicing

To enable the practical implementation of RAN in 5G, the management and orchestration of the system with respect to RAN slicing is driven by means of SDN. Our concept follows a hierarchical scheduling approach similar to the one presented in Section 2.3.2. In that setup, a master controller is based on the SD-RAN controller and it runs the inter-slice scheduler, whereas each slice is managed by a scheduling entity known as the **slice manager**. This is illustrated in Fig. 3.1. In turn, each slice manager operates the intra-slice scheduler, which was explained in Chapter 2.

The SD-RAN controller is the heart of the SD-RAN platform, which manages the interaction with the VMNOs or third parties and the slice managers. It receives requests from the slices/tenants from the northbound API. These requests are utilized as inputs for the algorithm described in the following and impact the resource allocation. The resource allocation policy entails which resources should be given to which slice and to which BS, as illustrated in Fig. 3.1. Within the SD-RAN controller we propose three main functionalities, which render an efficient RAN slicing.

Initially, an **application request processor** gathers and distributes the RAN slicing requests from the tenants to the **resource allocator** block. The resource allocator is in charge of dynamically re-assigning resources to slice managers within each BS. Finally, to enable the RAN slicing process, a **slice manager statistics** block contains all the information with respect to UEs within a slice and their respective channel conditions in the form of Channel Quality Indicator (CQI)s. The SD-RAN controller utilizes these statistics to improve the resource allocation process. The frequency of statistics' updates depends on the channel coherence time τ. It is important those messages to contain up-to-date statistics every time the channel changes in order to perform the best allocation. As already mentioned, updates are performed every 20 ms [MC08]. The SD-RAN controller dynamically re-assigns resources to each slice manager, whereas the latter is responsible for allocating them to UEs within that slice. The duty of the SD-RAN controller is to cooperate effectively with the slice managers in order to increase the efficiency of the network. For the remainder of this chapter, we adopt the principle of `FlexRAN` [Fou+16] and `5G-EmPOWER` [CKR19], and incorporate further functions to enable the proposed slicing approach. Further details are provided in the following subsections.

3.1.4 Related Work

The problem of RAN slicing has received a significant attention in the last years with vast ongoing research. Valuable conceptual works regarding RAN slicing can be found in [Ros+17a; SCS16; Fer+18; KN17; Li+20b]. From the practical perspective, the new concept of decoupling the control and data plane of conventional RANs facilitates the development of prototype SD-RAN platforms [FMK17; Fou+16; CKR19; SIN21], which provide a valuable asset for academia to deploy and test algorithms in realistic scenarios. However, only a limited number of UEs and BSs can be tested due to expensive radio equipment.

3.1.4.1 Single-BS Scenario

From the mathematical perspective, the RAN slicing problem has been addressed mainly for a single-BS scenario, ranging from optimizations [Man+19; Kok+11; Shi+18; YQC17; KLG; CNS18; KS18; Vil+20; SCN19], to genetic algorithms [HLS18], physical layer perspective [Pop+18] as well as the business model aspect [Beg+17; Sal+18]. Furthermore, deep reinforcement learning has been utilized in [Mei+21], which adapts to dynamic traffic demands and preserves long term QoS requirements.

From the industrial point view, the authors in [Gin+21] consider a network slicing use case under deterministic traffic assumptions, demonstrating higher throughput compared to alternative solutions. While the aforementioned works are valuable and are shown to be efficient for a single-BS scenario, they cannot be used for a multi-BS scenario, where the interference among adjacent BSs has to be considered carefully when allocating resources to slices.

3.1.4.2 Multi-BS Scenario

Similar to the scenario with a single-BS, there exist works that consider RAN slicing in a multi-BS scenario.

For instance, authors in [Shi+21] provide a QoS preservation scheduling approach for heterogeneous traffic requirements and prove to be efficient in achieving QoS. However, they do not specifically provide evaluations on a per-UE level QoS. In [SDC19], a RAN slicing solution is provided for multi-BSs while maximizing the spectral efficiency. However, the granularity remains on a slice level and no details about the UE throughput are provided. When a frequency selective wireless channel is considered, distinct UEs receive wireless resources in an unequal manner, depending on their CQIs.

The work in [Sal+17] presents different possibilities for the RAN slicing over multiple BSs and demonstrates a qualitative representation of the isolation effect of the approaches. Yet, QoS in terms of throughput is portrayed on a slice level and not on a UE level. Furthermore, the solutions provided in [Cab+17; Cab+19] consider a multi-BS scenario introducing an algorithm for dynamic resource allocation among slices. The authors in [Cab+17; Cab+19] compare their approach with two baselines solutions considering static slicing. But, they do not directly address the network slice isolation problem. Alternatively, the authors in [Sun+20] combine the UE admission control with the resource scheduling problem to achieve the UE QoS requirements such as delay and throughput. However, their optimization problem focuses on minimizing resource consumption while satisfying QoS requirements, rather than maximizing the overall system throughput.

The closest work to our approach is [DOr+19], where a method to enforce the network slicing policies of VMNOs over multi-BSs is proposed denoting slice isolation as the ability to provide a certain required percentage of resources to a slice. Whereas this might be sufficient for flat-fading wireless channels, where all the resources are the same for a UE, it might not be optimal in a frequency selective and time-variant case. In such scenarios selecting the best channel per-UE within a slice improves the network performance. On the contrary, if a bad channel is assigned to a network slice, even if the required percentage of resources is achieved, no UE QoS can be guaranteed. Motivated, by such scenarios, in this chapter, we propose a scheme which is more granular by offering a minimum achievable rate per-UE within a slice and demystifying results with respect to the slice isolation effect.

3.2 RAN Slicing Provisioning while Minimizing Network Resources

Network slicing has been targeted both on the CN [Ble+15; Qaz+17] and RAN level [KN17; LY15]. However, while network slicing on the CN is well investigated, in RAN it still remains an interesting and challenging problem due to the more complex resource coupling and the stochastic nature of the wireless channels. An intuitive way of providing a network slice is by allocating time-frequency resources in a persistent fashion. Yet, such a static approach does not take into account the stochastic nature of wireless channels and comes at the cost of loosing diversity gains. The latter stems from allowing other slices to use those resources if available. Thus, dynamic re-assignment of slices' resources has been proposed in the literature, up to the extreme case that they are re-assigned in each slot (i.e., 1 ms) [KLG; Kok+11]. This helps adapting to the dynamic changes of the wireless channel, but renders resource assignment more complex.

In this section, we account both for isolation on the wireless resource level and QoS performance, while enabling diversity gains. In that regard, our first contribution is to formulate and solve an optimization problem for resource usage reduction by dynamically re-assigning resources to slices, while providing a requested average rate and delay to each slice in an isolated fashion. For the remainder of this chapter, isolation refers to the ability of each slice to achieve a minimum requirement (i.e., delay, throughput) irrespective of the activity of the UEs in other slices. Initially, we consider the case of a single BS, where the problem is solved by the Lyapunov optimization [Nee10]. Realistic simulations are performed to show the effectiveness of our approach.

3.2.1 Problem Formulation

Initially, we demonstrate the case of a downlink cellular system with a single BS. Hence, inter-cell interference can be neglected and network slices are deployed within that BS, for which the equations from Section 3.1.1 as well as the SD-RAN solution of Section 3.1.3 hold.

For the first use case, we consider two types of slices. The first corresponds to capacity-critical slices, which require a minimum expected throughput. The second type corresponds to delay-critical slices, where a maximum expected delay must be guaranteed. Each slice contains a queue for the incoming slice traffic, which has a backlog of $U_s(t)$. The arrival traffic per slot is modeled by a random variable $\lambda_s(t)$ with mean value $\bar{\lambda}_s$, which denotes the amount of traffic in kbits/slot. The arrived traffic is accepted portion-wise after an admission control, with the admitted traffic at each slot denoted by the variable $\alpha_s(t)$. At each slot the admitted traffic is then served by a slice rate expressed by $r_s(t)$. Correspondingly, the slice queue backlog evolves from slot to slot according to the equation

$$U_s(t+1) = \max\left\{U_s(t) + \alpha_s(t) - r_s(t), 0\right\}, \quad \forall s \in \mathcal{S}. \tag{3.5}$$

Utilizing the expected queue backlog, the average delay of the queue can be calculated by using Little's theorem that is $D_s = U_s/r_s$ [XE13].

The SD-RAN controller collects the requests from the slices, namely $\underline{C}_s$, which denotes the minimum required throughput, as well as D_s^{max}, which is the maximum tolerable delay. Also, each slice specifies a rate policy $\{\gamma_i^s\}$ for $i \in \mathcal{N}$ to the controller, that provides information regarding the relative share of slice's minimum throughput among the slice's UEs as a percentage. Finally, the SD-RAN controller decides for a maximum throughput allowed for each slice, namely $\overline{C}_s$, depending on the wireless channel conditions. It is this maximum throughput that ensures rate isolation among the slices, as it prevents them from overloading the network. An intuitive way of selecting $\overline{C}_s$ is by assigning a portion of the channel capacity in an equal fashion to each slice.

We model the resource allocation as a minimization problem in terms of the utilized network resources, while guaranteeing slice QoS (i.e., delay and throughput), and slice isolation. Note that the notation of $b \in \mathcal{B}$ is omitted for a single-BS scenario in the equations, since only one BS exists in the system. The optimization problem is formulated as:

$$\min_{w_{i,j}(t),\alpha_s(t)} \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \sum_{s=1}^{S} \sum_{i=1}^{N} \sum_{j=1}^{R} w_{i,j}^{s}(t) \tag{3.6a}$$

$$\text{s.t.} \quad \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \alpha_s(t) \leq \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} r_s(t), \quad \forall s \in \mathcal{S}, \tag{3.6b}$$

$$\lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} U_s(t) \leq D_s^{max} \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} r_s(t), \quad \forall s \in \mathcal{S}, \tag{3.6c}$$

$$\lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} r_i(t) \geq \gamma_i^s \cdot \underline{C}_s \cdot \delta_i^s, \quad \forall i \in \mathcal{N}, \forall s \in \mathcal{S}, \tag{3.6d}$$

$$\lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} r_s(t) \leq \overline{C}_s, \quad \forall s \in \mathcal{S}, \tag{3.6e}$$

$$\sum_{s=1}^{S} \sum_{i=1}^{N} w_{i,j}^{s}(t) \leq 1, \quad \forall j \in \mathcal{R}, \forall t \in \mathcal{T}, \tag{3.6f}$$

$$0 \leq \alpha_s(t) \leq \lambda_s(t), \quad \forall s \in \mathcal{S}, \forall t \in \mathcal{T}, \tag{3.6g}$$

$$w_{i,j}(t) \in \{0,1\}, \quad \forall i \in \mathcal{N}, \forall j \in \mathcal{R}, \forall t \in \mathcal{T}. \tag{3.6h}$$

The minimization problem given in (3.6a) targets the allocation of the minimum number of resources to satisfy the constraints. This is especially important, since by minimizing the amount of utilized resources, we can accommodate more slices in the network. Constraint (3.6b) ensures that the average throughput of each slice is greater than its average admitted traffic. Constraint (3.6c) guarantees that the average delay will not surpass an average maximum set delay D_s^{max}, where $U_s(t)$ represents the queue backlog of slice s in slot t. Furthermore, constraint (3.6d) guarantees that the average throughput of each slice's UE is greater than the share of minimum throughput defined by policy γ_i^s, while constraint (3.6e) ensures that the average throughput of each slice will not exceed a maximum throughput set in order to provide isolation among slices. Additionally, constraint (3.6f) defines the orthogonality constraint, that a PRB can be allocated to at most one UE at a time and finally, (3.6g) suggests that the total admitted traffic cannot exceed the arrival traffic in any slot. Note that the decision variable $w_{i,j}(t)$ is binary.

3.2.2 Resource Allocation with Lyapunov Optimization

The aforementioned problem is a stochastic optimization problem, which is hard to solve due to the unknown channel variations and incoming traffic over the slots. Even with full knowledge, the complexity increases when a large T is considered. In this case, linear programming techniques can not be used to provide a solution to the problem. Therefore, the Lyapunov optimization approach [Nee10] is used to solve the problem (3.6a) - (3.6h). The constraints are transformed into virtual queues, which evolve over slots according to a process described by:

$$U_s(t+1) = \max\left\{U_s(t) + \alpha_s(t) - r_s(t), 0\right\}, \quad \forall s \in \mathcal{S}, \tag{3.7}$$

$$G_s(t+1) = \max\left\{G_s(t) + U_s(t) - D_s^{max} \cdot r_s(t), 0\right\}, \quad \forall s \in \mathcal{S}, \tag{3.8}$$

$$K_i(t+1) = \max\left\{K_i(t) + \gamma_i^s \cdot \underline{C}_s \cdot \delta_i^s - r_i(t), 0\right\}, \quad \forall i \in \mathcal{N}, \forall s \in \mathcal{S}, \tag{3.9}$$

$$J_s(t+1) = \max\left\{J_s(t) + r_s(t) - \overline{C}_s, 0\right\}, \quad \forall s \in \mathcal{S}. \tag{3.10}$$

The virtual queue $U_s(t)$ is expressed by (3.6b) and in our scenario corresponds to the physical queue denoted by (3.5), whereas virtual queues $G_s(t)$, $K_i(t)$ and $J_s(t)$ represent the constraints presented in (3.6c)-(3.6e) respectively. Defining $K_s(t) = \left[..., K_i^s(t), ...\right]$ for $i \in \mathcal{N}$, the overall system queue state is denoted by $\Theta(t) = \{\Theta_s(t)\}$, where $\Theta_s(t) = [G_s(t), U_s(t), K_s(t), J_s(t)]$ is the state of slice s. We now define the quadratic Lyapunov function as

$$L(\Theta(t)) = \frac{1}{2}\sum_{s\in\mathcal{S}}(J_s(t))^2 + \frac{1}{2}\sum_{s\in\mathcal{S}}(U_s(t))^2 + \frac{1}{2}\sum_{s\in\mathcal{S}}(G_s(t))^2 + \frac{1}{2}\sum_{s\in\mathcal{S}}\sum_{i\in\mathcal{N}}(K_i(t))^2. \tag{3.11}$$

According to the Lyapunov optimization, the Lyapunov function is a scalar metric that measures the *queue congestion* state. This function is always non-negative and equals to zero only in the case when $\Theta(t) = 0$ (when the queues have reached their stability). When $L(\Theta(t))$ is small, the virtual queues are also small. Thus, a low queue congestion and a high stability is implied. In turn, a large $L(\Theta(t))$ indicates high congestion and low system stability. In any case, all constraints are satisfied if, and only if, the infinite time-horizon limit of $L(\Theta(t))$ is bounded, i.e., $\lim_{T\to\infty} 1/T \sum_{t=0}^{T-1} L(\Theta(t)) < \infty$. Next, the Lyapunov drift [Nee10] is introduced and denoted by

$$\Delta L(\Theta(t)) = \mathbb{E}\left\{L(\Theta(t+1)) - L(\Theta(t)) | \Theta(t)\right\}. \tag{3.12}$$

Based on the Lyapunov technique, the objective is to minimize an infinite bound on the Lyapunov drift-plus-penalty in each time slot, where the drift-plus-penalty is expressed as

$$\Delta L(\Theta(t)) + V\mathbb{E}\left\{\sum_{s=1}^{S}\sum_{i=1}^{N}\sum_{j=1}^{\mathcal{R}} w_{i,j}(t)\cdot\delta_i^s|\Theta(t)\right\}. \tag{3.13}$$

The design parameter $V \geq 0$ is used to determine how much emphasis is put on resource minimization, compared to the queue stability. The drift-plus-penalty expression elaborated in (3.13) satisfies the expression stated in

$$\Delta L(\Theta(t)) + V\mathbb{E}\left\{\sum_{s=1}^{S}\sum_{i=1}^{N}\sum_{j=1}^{\mathcal{R}} w_{i,j}(t)\cdot\delta_i^s\right\} \leq B + V\mathbb{E}\left\{\sum_{s=1}^{S}\sum_{i=1}^{N}\sum_{j=1}^{\mathcal{R}} w_{i,j}(t)\cdot\delta_i^s\right\}$$

$$+\mathbb{E}\left\{\sum_{s\in\mathcal{S}}\left(r_s(t)-\overline{C}_s\right)J_s(t)+\sum_{s\in\mathcal{S}}\left(\alpha_s(t)-r_s(t)\right)U_s(t)\right\} \tag{3.14}$$

$$+\mathbb{E}\left\{\sum_{s\in\mathcal{S}}\left(U_s(t)-D_s^{max}\cdot r_s(t)\right)G_s(t)+\sum_{i\in\mathcal{N}}\left(\gamma_i^s\cdot\underline{C}_s\cdot\delta_i^s-r_i(t)\right)K_i(t)\right\},$$

where $B > 0$ is a constant. We refer the readers to [Nee10, Lemma 4.6] for the full proof of the above expression. In order to solve the stochastic problem given in (3.6a) - (3.6h) we minimize the right-hand-side of (3.14) in every time slot, subject to the constraints (3.6f)-(3.6h).

Algorithm 1 Admission Control

```
1: Collect the status of the admission control queue length $U_s(t)$,   $\forall s \in S$
2: if $U_s(t) = 0$ then
3:     Admit $\alpha_s(t) = \lambda_s(t)$ to minimize (3.15)
4: else
5:     Admit $\alpha_s(t) = 0$ to minimize (3.15)
6: end if
```

As can be seen from (3.14), the statement on the right-hand-side which we refer to as L, is convex as the function contains a linear form of the decision variable. Therefore we can find its minimum efficiently. In order to do so, we calculate the derivative with respect to the two decision variables, namely resource allocation $w_{i,j}(t)$ and admission control $\alpha_s(t)$, and distribute the resources accordingly:

$$\frac{\partial L}{\partial \alpha_s(t)} = U_s(t) \geq 0, \tag{3.15}$$

$$\frac{\partial L}{\partial w_{i,j}(t)} = V - r_{i,j}(t)\left(G_s(t) - J_s(t) + U_s(t) + K_i(t)\right). \tag{3.16}$$

Algorithm 2 Dynamic resource assignment

```
1: Initialization:
2: Choose $V > 0$;
3: Set $G_s(0) = J_s(0) = U_s(0) = K_i(0) := 0 \; \forall i \in \mathcal{N}, \forall s \in S$
4: for $t := 1, 2, 3, \ldots, \infty$ do
5:     Slice Manager:
6:     Calculate $v_{s,j}(t) = \max_{i \in \mathcal{N}} \left\{ r_{i,j}(t) \cdot \delta_i^s \left[ G_s(t) - J_s(t) + U_s(t) + K_i(t) \right]^+ \right\}$
7:     Store $n_{s,j}(t) = \arg\max_{i \in \mathcal{N}} \left\{ r_{i,j}(t) \cdot \delta_i^s \left[ G_s(t) - J_s(t) + U_s(t) + K_i(t) \right]^+ \right\}$
8:     Communicate $v_s(t) = \left\{ v_{s,j}(t) \right\}$ to the SD-RAN Controller
9:     SD-RAN Controller:
10:    Assign resource $j$ to slice $s_j(t) = \arg\max_{s \in \mathcal{S}} \{ v_s(t) \geq V \}$
11:    Slice Manager:
12:    If resource $j$ was allocated, assign it to $n_{s,j}(t)$.
13:    Update $G_s(t), U_s(t), K_i(t), J_s(t)$, based on (3.8), (3.7), (3.9), (3.10).
14: end for
```

According to (3.15), we derive the statement on the right-hand-side of (3.14), i.e., L, with respect to the decision variable of the admitted traffic of each slice. As minimization is the target, the result is that $\alpha_s(t) = 0$, i.e., traffic should not be accepted when $U_s(t) > 0$, while it may be accepted arbitrarily if $U_s(t) = 0$. This induces an admission control procedure that we state in Alg. 1. In order to apply the dynamic resource allocation, due to the orthogonality constraint (3.6f), each PRB should optimally be given to the UE with the maximum value of $r_{i,j}(t) \cdot \delta_i^s \left[G_s(t) - J_s(t) + U_s(t) + K_i(t) \right]$, as long as it is larger than V. Alternatively, it should not be assigned at all if all values are lower than V.

The optimization is implemented by using a two-stage bidding system, as shown in Alg. 2. Each slice manager maintains the queue states $\Theta_s(t) = [G_s(t), U_s(t), K_s(t), J_s(t)]$ of its slice and updates them from slot to slot. Further, it calculates the value

$$v_{s,j}(t) = \max_{i \in \mathcal{N}} \left\{ r_{i,j}(t) \cdot \delta_i^s |G_s(t) - J_s(t) + U_s(t) + K_i(t)| \right\}, \tag{3.17}$$

that each PRB is worth to it, stores the associated UE $n_{s,j}(t) \in \mathcal{N}$ that belongs to slice s and communicates $v_{s,j}(t)$ to the SD-RAN controller. The latter then matches the values and assigns each resource to the slice manager to which it is of most value, or to none of it is smaller than V. The slice manager then schedules the resources internally to the UE which produced this value.

3.2.3 Performance Evaluation

In the evaluation part, the effectiveness of the proposed method is investigated with respect to satisfying the QoS requirements, while also providing isolation guarantees. The design parameter V of the Lyapunov technique introduces the trade-off between the resource utilization

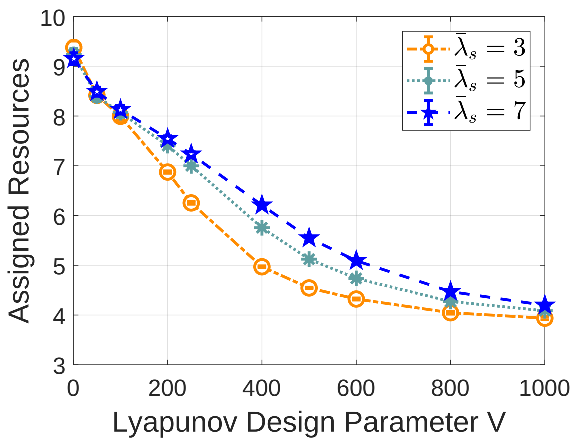


Figure 3.2: The number of assigned resources with respect to the design parameter V given a maximum threshold $\overline{C}_s$ of 7 kbits/slot per each slice, evaluated for various traffic arrivals.

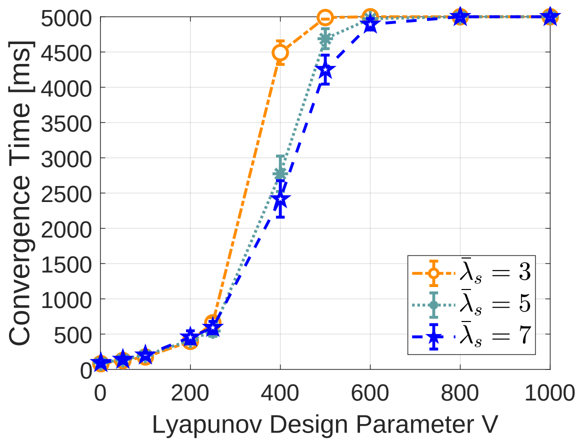


Figure 3.3: The convergence time of the optimization with respect to the design parameter V given a maximum threshold $\overline{C}_s$ of 7 kbits/slot per each slice, evaluated for various traffic arrivals.

and the queue stability convergence time, which defines the time it takes for the optimization to satisfy the constraints and stabilize. Therefore, the first analysis consists of determining the correct value of V for the given scenario. All the simulations are repeated 100 times and the confidence intervals are derived in order to correctly evaluate the performance. The overall simulation parameters are based on Table 3.2 presented in Section 3.1.1, where the input requirements for network slices are presented in Table 3.3.

Table 3.3: Network slices' input parameters

Slice Parameters	**Values**
Number of slices	2
Number of users per slice	5
Minimum throughput required per slice	4 kbits/slot
Delay-critical slice maximum tolerable delay	10 ms
Maximum allowed throughput per slice	7 kbits/slot

3.2.3.1 Optimality and Algorithm Convergence Time

Initial results are presented with respect to the resource minimization objective and the convergence time of the proposed approach with respect to the design parameter V of the Lyapunov technique in Fig. 3.2 and Fig. 3.3, respectively. Fig. 3.2, illustrates the number of PRBs needed to satisfy the QoS requirements of the slices depending on different design parameter values and traffic demands. As denoted by Fig. 3.2, the number of the assigned resources increases with the incoming traffic $\bar{\lambda}_s$ and decreases with the design parameter

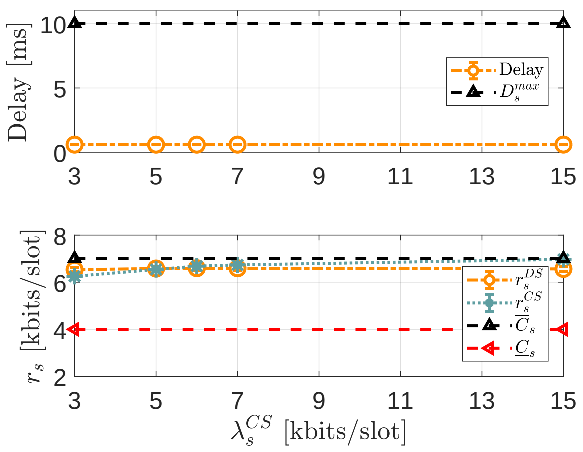


Figure 3.4: Average throughput of the delay-critical and throughput-critical slices as well as average delay of the delay-critical slice with respect to various traffic demands of the throughput-critical slice given a 5 kbits/slot arrival of the delay-critical slice and $V = 250$.

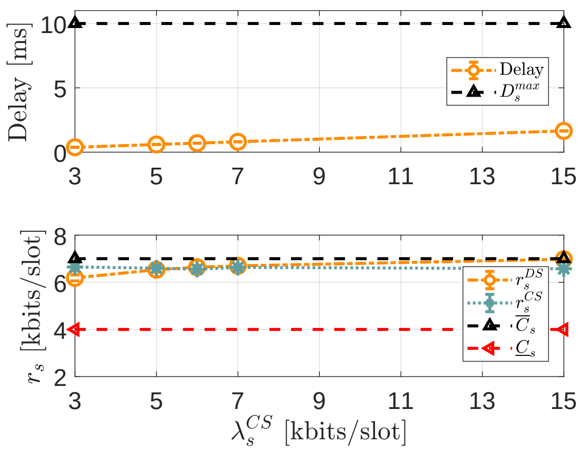


Figure 3.5: Average throughput of the throughput-critical and delay-critical slice as well as the average delay of the delay-critical slice with respect to various traffic demands of the delay-critical slice given a 5 kbits/slot arrival of the throughput-critical slice and $V = 250$.

value V. In any case, the algorithm manages to satisfy the arrivals, as long as they do not overload the network. However, the larger V is, the closer to the optimal resource utilization the algorithm is, as the number of assigned PRBs to achieve the requirements decreases.

Alternatively, Fig. 3.3 illustrates the queue stability convergence time with respect to various arrivals and values of V. As depicted, the convergence time increases with the design parameter value V, since the method assigns resources less frequently and therefore a larger time is needed for the queues to reach a stable state. Interestingly, the larger the traffic arrival is, the lower the convergence time. Due to the fact that the queue state Θ_s increases faster for larger arrivals, the queues reach the required steady-state magnitude faster.

Given the results from Fig. 3.2 and Fig. 3.3, a trade-off regarding the queue stability convergence time and the resource utilization must be decided, such that the system can be evaluated correctly. Again, we stress that regardless of the chosen Lyapunov design parameter V, the constraints are not violated, only the time required to satisfy them increases. According to the observations from the aforementioned figures, a suitable design parameter value is $V = 250$, which is therefore considered for all the upcoming evaluations of this section, if not stated otherwise.

3.2.3.2 Isolation Performance

Isolation performance in network slicing can be evaluated by the impact of changes happening in other slices, such as the number of UEs or traffic demands, have on a given slice. In order to correctly measure and evaluate the isolation performance given by the introduced model, we consider two scenarios: 1) Given a traffic demand for the delay-critical slice, the traffic demand of the throughput-critical slice is varied to evaluate the influence on the achieved throughput; 2) Similarly, given a traffic demand for the throughput-critical slice, the traffic demand of the delay-critical slice is varied to evaluate the influence on the maximum delay.

Fig. 3.4 illustrates the delay-critical slice achieved throughput, referred to as r_s^{DS}, for different variations of the throughput-critical slice traffic arrivals, λ_s^{CS}. Given an incoming traffic demand of 5 kbits/slot for the delay-critical slice, the incoming traffic arrival of the throughput-critical slice varies between 3-15 kbits/slot, where $\lambda_s^{CS} = 3$ kbits/slot is lower than $\underline{C}_s$, whereas $\lambda_s^{CS} = 15$ kbits/slot is larger than $\overline{C}_s$. As denoted by Fig. 3.4, the achieved throughput of the delay-critical slice, namely r_s^{DS} is not affected by the change of the traffic arrival of the other slice. Moreover, the average delay for the delay-critical slice is shown to be unaffected and remains below the requirement D_s^{max}. As for the throughput of the throughput-critical slice referred to as r_s^{CS}, it remains larger than the minimum requirement $\underline{C}_s$ and regardless of the traffic demand does not exceed the maximum threshold $\overline{C}_s$. Therefore, the isolation among the two slices can be maintained.

Similarly, Fig. 3.5 illustrates the outcomes of the second scenario, where the traffic arrival of the investigated throughput-critical slice is fixed to 5 kbits/slot, whereas the traffic arrival of the delay-critical slice varies in the range of 3-15 kbits/slot. Even in this case, as shown in Fig. 3.5, the traffic variation of the delay-critical slice referred to as λ_s^{DS}, does not affect at all the achieved throughput of the throughput-critical slice r_s^{CS}. Moreover, the minimum required throughput is achieved for the delay-critical slice, and further, the maximum threshold is not exceeded regardless of the traffic arrival. Finally, the average delay does not exceed the maximum tolerable delay D_s. These results indicate the effectiveness of our approach in the ability to achieve the QoS requirements, while guaranteeing a smooth functionality and isolation among the slices.

3.2.3.3 Maximum Threshold Selection

Further investigations are conducted to demonstrate the impact of the maximum threshold on the resource utilization. As shown in Fig. 3.6, the resource assignment increases with $\overline{C}_s$. Moreover, by varying the values of the Lyapunov design parameter we can see that the higher the V, the more stable the system gets and the global minimum on the resource utilization is achieved. Therefore, we can conclude that the design parameter V and the maximum threshold $\overline{C}_s$ are coupled and a careful selection has to be made depending on the incoming traffic. In general, the selection of a maximum threshold $\overline{C}_s$ above the network capabilities

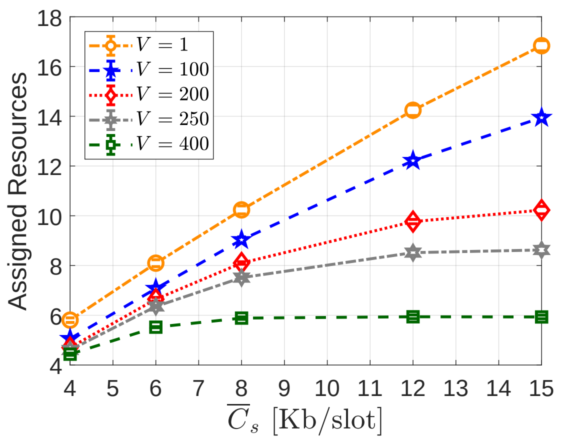


Figure 3.6: Number of assigned resources with respect to the maximum throughput threshold $\overline{C}_s$ per slice given a traffic arrival of 5 kbits/slot for each slice.

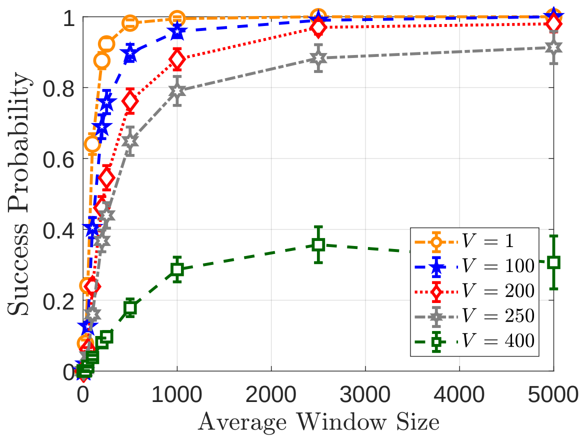


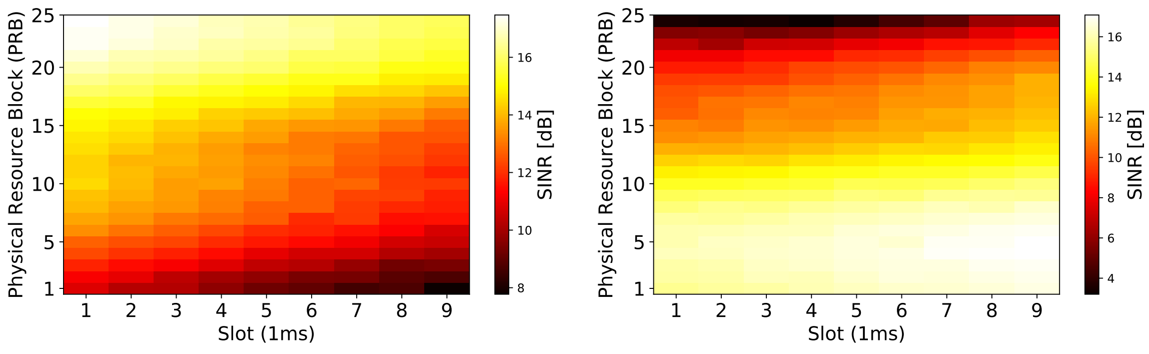
Figure 3.7: Probability of achieving the QoS requirements with respect to various average window sizes for various V values given a traffic arrival of 5 kbits/slot and $\overline{C}_s$ of 7 kbits/slot per slice.

cannot guarantee the isolation among the slices, as the network would not be able to process all the incoming traffic.

Given the fact that the Lyapunov technique operates on expected values, instantaneous rates and delays are in fact not formally guaranteed with this approach. However, we demonstrate the probability of achieving the QoS requirements in a given, small-time window. In Fig. 3.7, the probability of satisfying all QoS requirements within a finite, moving average window size is depicted for various design parameter values V. As shown, a lower value of V allows the algorithm to converge faster and given a window size of 500 slots it satisfies the requirements with a probability which is close to 1, for $V = 1$. Moreover, a decreasing trend on the probability is noticed when the V parameter becomes larger. Particularly, for $V = 400$, even if the average window size corresponds to the entire simulation time of 5000 slots, the probability of achieving the required QoS is low. The reason for this behavior relates to the emphasis on the resource utilization, when a larger V is selected. Consequently, we observe the importance of the trade-off between reaching the optimum and the time required to reach the queue stability when selecting the Lyapunov parameter V accordingly.

3.3 User-Based Quality of Service Aware Multi-BS Radio Access Network Slicing

While RAN slicing offers more efficient utilization of spectrum resources [Ros+17a; SCS16], it renders slice isolation challenging, especially in a wireless environment due to its inherent stochastic nature and scarce wireless resources. So far, the problem of slice isolation has been



(a) t=0 ms (b) t=240 ms

Figure 3.8: Received Signal-to-Interference-Plus-Noise-Ratio (SINR) of one UE for one LTE frame (i.e., 10 slots) in 5 MHz bandwidth at different time instances in low Doppler frequency Extended Pedestrian A (EPA) channel.

mainly tackled for a single-BS scenario [Man+19; Kok+11]. However, in reality a cellular network consists of multi-BSs. Considering a multi-BS scenario, the slice isolation preservation becomes even more challenging due to the limited available frequency bands [Fed], resulting in an increased interference introduced by slices deployed in adjacent BSs. There already exist interference management techniques in 5G [Nam+14; Hos+14; Jun+14] with respect to the multi-BS scenarios, but none of them considers the problem of RAN slicing. The latter is more challenging especially due to the lack of information shared among different network slices operated individually by VMNOs [DOr+19].

When considering slice isolation, in general, there are two questions that need to be answered: *1) What is the right metric for isolation? and 2) What should be the isolation granularity: slice based or user based?* Regarding the first question, the literature is divided in two main categories: radio resource-based isolation [DOr+19; KLG], i.e., if a slice is provided with a specific amount of resources, and performance-based isolation [Man+19; Kok+11; Cab+17] i.e., if a slice fulfills a minimum QoS requirement. While the former would be sufficient for flat-fading wireless channels, in reality it would be more beneficial to consider the latter, as it takes into account the QoS requirements and therefore can benefit from diversity gains in case of frequency-selective wireless channels. Considering the second question, the state of the art is also divided in two groups. For instance, there are works that define the radio resource-based and performance-based isolation mainly on the slice level [SDC19; CNS18; KS18] and works based on the UE level [Man+19; Kok+11; Pap+19b]. We stress that while the first approach provides QoS for each UE when UEs depict homogeneous characteristics, it cannot provide user QoS if UEs experience different channel characteristics. We conclude that in order to guarantee UE QoS, an isolation at UE level is mandatory.

To illustrate the above points, let us consider Fig. 3.8. In the figure, the wireless channel is depicted in terms of received SINR for one UE at different time instances in a low mobility environment i.e., 5 Hz Doppler frequency EPA channel [3GP08] in 5 MHz bandwidth corresponding to 25 PRBs. The SINR is displayed as a colorbar where the x-axis represents time in slots and the y-axis denotes the number of PRBs. Considering the relation between SINR and reliability, it is apparent that due to the frequency selective fading, the allocation of PRBs between 1 and 5 result in a different throughput than an allocation of PRBs between 20 and 25, although for one UE, 20% of the resources would have been assigned in both cases. Furthermore, the SINR and frequency-time grid relation changes over time due to UE mobility. Fig. 3.8a and Fig. 3.8b illustrate the same wireless channel model realization, but with 240 ms time difference. Thus, both frequency and time selectivity of the channel should be considered in the resource allocation optimization. In fact, in [MLG06] it is shown that wireless channel dependent scheduling can increase the achieved throughput up to 80% compared to static scheduling. Hence, approaches that tend to neglect the wireless channel effects diminish the effectiveness of the RAN slicing allocation.

Given all the above mentioned issues, in this section we provide the definition of slice isolation on a per-UE basis. This is necessary to provide actual guarantees in the network, while considering a multi-BS scenario. Specifically, our main contributions are:

- We propose a RAN slicing isolation definition based on performance guarantees on a per-UE basis to cater for frequency selective wireless channels and time variations.
- We provide a mathematical formulation for RAN slicing and isolation in a multi-BS scenario with slices being deployed in adjacent interfering BSs with focus on throughput maximization, while satisfying UE constraints. Due to the complexity of the problem, we propose an approximation solution based on Lyapunov optimization.
- We demonstrate the effectiveness of our approach with respect to UEs and BSs as well as in terms of the convergence time of the algorithm. To verify our results, we consider an aircraft in-cabin channel model representing a 5G multi-BS use case with high UE density.
- We compare our algorithm with existing state-of-the-art solutions with respect to network slice isolation and throughput maximization, as well as CPU, memory utilization and communication overhead.

3.3.1 Problem Formulation

Here, we present the architectural concept of our solution which is based on the principles of SD-RAN as elaborated in [FMK17; Fou+16; CKR19] with the help of Fig. 3.1. Initially, we detail our envisioned SD-RAN solution. Then, we describe the interaction of VMNOs or third parties with the SD-RAN controller utilizing RAN slicing requests. Finally, we present the algorithm to perform RAN slicing which optimizes the system performance.

3.3.1.1 RAN Slicing Request

Each slice $s \in \mathcal{S}$ communicates to the SD-RAN controller its requests through the northbound API. As explained earlier in this chapter, these requests are expressed either in terms of minimum radio resource requirements or strict QoS requirements that need to be satisfied. These can either be expressed on a per-UE level, or on a per-slice level. Let $\rho_s \in [0, 1]$ express the ratio of resources requested by slice s such that $\sum_{s \in \mathcal{S}} \rho_s = 1$. Each slice has an average traffic arrival $\bar{\lambda}_s$ rate per slot, whereas the system admits traffic portion-wise with an average rate α_s per slot. Further, let $K_s(t) = \sum_{b=1}^{B} \sum_{i=1}^{N} \sum_{j=1}^{R} w_{i,j}^{b,s}(t)$ be the total number of PRBs allocated to slice $s \in \mathcal{S}$ in slot $t \in \mathcal{T}$. Finally, let R be the total amount of PRBs per BS and η_s^b a binary variable that denotes whether a slice is allowed to be served by BS $b \in \mathcal{B}$ or not. Note here that η_s^b is different from δ_i^s, where the latter identifies whether a UE $i \in \mathcal{N}$ belongs to a slice $s \in \mathcal{S}$.

Definition 1 *(Aggregate radio resource requirement) Given a minimum radio resource requirement per slice ρ_s, network slices are considered isolated if:*

$$\lim_{T \to \infty} \frac{1}{T} \sum_{t=0}^{T-1} K_s(t) \geq \rho_s \cdot R \cdot \sum_{b=1}^{B} \eta_s^b. \tag{3.18}$$

Definition 2 *(QoS requirement per slice) Given a minimum QoS rate per slice $\underline{C_s}$, network slices are considered isolated if:*

$$\lim_{T \to \infty} \frac{1}{T} \sum_{t=0}^{T-1} \sum_{b=1}^{B} \sum_{i=1}^{N} \sum_{j=1}^{R} w_{i,j}^{b,s}(t) \cdot r_{i,j}^{b}(t) \geq \underline{C_s}, \quad \forall s \in \mathcal{S}. \tag{3.19}$$

Definition 3 *(QoS requirement per UE) Given a minimum QoS rate per UE $i \in \mathcal{N}$, denoted as $\underline{C_s^i}$, network slices are considered isolated if*

$$\lim_{T \to \infty} \frac{1}{T} \sum_{t=0}^{T-1} \sum_{b=1}^{B} \sum_{j=1}^{R} w_{i,j}^{b,s}(t) \cdot r_{i,j}^{b}(t) \geq \underline{C_s^i}, \quad \forall i \in \mathcal{N}, \forall s \in \mathcal{S}. \tag{3.20}$$

3.3.1.2 RAN Slicing Problem Formulation

To maximize the number of UEs served in a slice, in our approach, the RAN slicing algorithm aims at maximizing the overall slice throughput. We follow Definition 3 explained above for assuring slice isolation, where we guarantee a minimum throughput requirement for the slice's UEs. The optimization problem is defined as follows:

$$\mathcal{P}_0 : \max_{w^b_{i,j}(t)} \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \sum_{b=1}^{B} \sum_{s=1}^{S} \sum_{i=1}^{N} \sum_{j=1}^{R} w^{b,s}_{i,j}(t) \cdot r^b_{i,j}(t) \tag{3.21a}$$

$$\text{s.t.} \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \alpha_s(t) \leq \lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} r_s(t), \quad \forall s \in \mathcal{S}, \tag{3.21b}$$

$$\lim_{T\to\infty} \frac{1}{T} \sum_{t=0}^{T-1} \sum_{b=1}^{B} \sum_{j=1}^{R} w^{b,s}_{i,j}(t) \cdot r^b_{i,j}(t) \geq \underline{C^i_s}, \quad \forall i \in \mathcal{N}, \forall s \in \mathcal{S}, \tag{3.21c}$$

$$\sum_{s=1}^{S} \sum_{i=1}^{N} \sum_{j=1}^{R} w^{b,s}_{i,j}(t) \leq R, \quad \forall t \in T, \forall b \in \mathcal{B}, \tag{3.21d}$$

$$\sum_{b=1}^{B} \eta^b_s \leq B_s, \quad \forall s \in \mathcal{S}, \tag{3.21e}$$

$$\sum_{s=1}^{S} \sum_{i=1}^{N_s} w^{b,s}_{i,j}(t) \leq 1, \quad \forall t \in T, \forall b \in \mathcal{B}, \forall j \in \mathcal{R}, \tag{3.21f}$$

$$0 \leq \alpha_s(t) \leq \lambda_s(t), \quad \forall t \in T, \forall s \in \mathcal{S}, \tag{3.21g}$$

$$w^b_{i,j}(t) \in \{0, 1\}, \tag{3.21h}$$

Constraint (3.21b) ensures that the average rate of each slice is greater than the average admitted traffic for each slice. In order to preserve equality among UEs within a slice and satisfy each UE's requirements, we introduce constraint (3.21c), which guarantees that the average rate of each UE within a slice is greater than a minimum threshold assigned by the slice owner i.e., $\underline{C^i_s}$. The maximum number of PRBs for each BS is R. This is equivalent to saying that all UEs that are served from different slices pertaining to BS $b \in \mathcal{B}$ cannot receive more than R PRBs in total. This is achieved by constraint (3.21d). Given that slices are distributed over multi-BSs, we define the variable η^b_s as a binary variable being 1 if a slice can be accommodated by BS b or 0 otherwise. We can limit the number of BSs which can share the same slice $s \in \mathcal{S}$, by using constraint (3.21e). The orthogonality constraint for each PRB is finally guaranteed by constraint (3.21f). Constraint (3.21g) suggests that the total admitted traffic cannot exceed the average arrival rate of each slice in order to operate in a stable region. Note that the decision variable $w^b_{i,j}(t)$ is binary.

The main challenge of optimally solving $\mathcal{P}_0$ stems from the stochastic nature of the problem setup. Namely, the lack of complete knowledge of the wireless channel as well as UE traffic requests over time makes the problem difficult to solve with traditional linear programming methods. Even with knowledge about the future events, with increasing size of the window for which we look at the problem at (i.e., T), the latter is hard to solve. The aforementioned challenges indicate that in order to solve $\mathcal{P}_0$, an online method that can provide a suboptimal solution based only on per slot input parameter knowledge. The Lyapunov optimization has been suggested and proven efficient for the aforementioned problems [Nee10]. Thus, we utilize it in our work to solve $\mathcal{P}_0$.

3.3.2 Approximation Solution with Lyapunov Optimization

As slice isolation is one of the main challenges in RAN slicing, a technique to obtain the isolation constraint as part of the objective function is needed. Following the Lyapunov optimization approach, the constraints are transformed into virtual queues and they become part of the objective function. The virtual queues of the problem evolve over time as:

$$U_s(t+1) = \max\{U_s(t) + \alpha_s(t) - r_s(t), 0\}, \quad \forall s \in \mathcal{S}. \tag{3.22}$$

$$L_i(t+1) = \max\{L_i(t) + \underline{C_s^i} - \sum_{b=1}^{B}\sum_{j=1}^{R} w_{i,j}^{b,s}(t)\cdot r_{i,j}^{b}(t), 0\}, \quad \forall i \in \mathcal{N}, \forall s \in \mathcal{S}. \tag{3.23}$$

The virtual queue $U_s(t)$ is a slice-based queue that indicates the physical backlog queue of the system, which evolves according to the admitted traffic and the served traffic on every slot $t \in \mathcal{T}$. Alternatively, the virtual queue $L_i(t)$ represents the UE queue that is related to the UE minimum throughput requirement stated by constraint (3.21c) in $\mathcal{P}_0$. Using the virtual queues we define an overall system queue state $\Theta(t) = \{\Theta_s(t)\}$ where $\Theta_s^t = \{U_s(t), L_i(t)\}$ indicates the state of slice $s \in \mathcal{S}$. The quadratic Lyapunov function is then defined as:

$$L(\Theta(t)) = \frac{1}{2}\sum_{s\in S}(U_s(t))^2 + \frac{1}{2}\sum_{s\in S}\sum_{i\in N}(L_i(t))^2, \quad \forall t \in \mathcal{T}. \tag{3.24}$$

The Lyapunov function is a scalar metric to measure the state of the *queue congestion*. A small value of $L(\Theta^t)$ implies that the stability of the queues holds and therefore the constraints are satisfied. In the contrary, if the algorithm cannot satisfy the constraints, then a large value of $L(\Theta(t))$ is observed and as a result we conclude that the system is not stable. In any case, all constraints are satisfied if, and only if, the infinite time-horizon limit of $L(\Theta(t))$ is bounded, i.e., $\lim_{T\to\infty} 1/T \sum_{t=0}^{T-1} L(\Theta(t)) < \infty$. According to [Nee10], to check for stability, we need to define first the Lyapunov drift of the Lyapunov functions in every slot $t \in \mathcal{T}$ as

$$\Delta L(\Theta(t)) = \mathbb{E}\left\{L(\Theta(t+1)) - L(\Theta(t))|\Theta(t)\right\}. \tag{3.25}$$

Based on the Lyapunov technique, the objective is to minimize an infinite bound on the Lyapunov drift in each time slot, where the drift-plus-penalty is expressed as

$$\Delta L(\Theta(t)) - V\mathbb{E}\{\sum_{b=1}^{B}\sum_{s=1}^{S}\sum_{i=1}^{N}\sum_{j=1}^{R} w_{i,j}^{b,s}(t)\cdot r_{i,j}^{b}(t)|\Theta(t)\}. \tag{3.26}$$

The parameter $V \geq 0$ is the Lyapunov design parameter, which controls the emphasis given to the maximization problem compared to the queue stability. That means, a larger V pushes the algorithm towards optimality, but increases the time needed for queues to converge. In our case, this is directly translated to the ability of users receiving the minimum required throughput (i.e., to preserve slice isolation). Referring to [Nee10, Lemma 4.6], the objective is to minimize a bound on the drift-plus-penalty expression that satisfies the constraint

$$\begin{aligned}&\Delta L(\Theta(t)) - V\mathbb{E}\{\sum_{b=1}^{B}\sum_{s=1}^{S}\sum_{i=1}^{N}\sum_{j=1}^{R} w_{i,j}^{b,s}(t)\cdot r_{i,j}^{b}(t)|\theta(t)\} \leq B\\ &- V\mathbb{E}\{\sum_{b=1}^{B}\sum_{s=1}^{S}\sum_{i=1}^{N}\sum_{j=1}^{R} w_{i,j}^{b,s}(t)\cdot r_{i,j}^{b}(t)|\theta(t)\} + \mathbb{E}\{\sum_{s\in S}(\alpha_s(t) - r_s(t))U_s(t)\}\\ &+ \mathbb{E}\{\sum_{i\in N}(\underline{C_s^i} - \sum_{b=1}^{B}\sum_{j=1}^{R} w_{i,j}^{b,s}(t)\cdot r_{i,j}^{b}(t))L_i(t)\}.\end{aligned} \tag{3.27}$$

Our proposed RAN slicing approach maximizes the right-hand-side of (3.27), subject to the constraints (3.21d) - (3.21h).

Given that the right-hand-side of (3.27) involves both linear and integer unknowns, we cannot prove convexity unless the decision variable $w_{i,j}^{b}(t)$ is relaxed and is assumed that it can take values such that $w_{i,j}^{b}(t) \in [0,1]$. Due to the fact that the expression on the right-hand-side of (3.27) and the constraints (3.21d) - (3.21h) are convex with respect to the relaxed continuous decision variable $w_{i,j}^{b}(t)$, we derive the expression with respect to the resource allocation $w_{i,j}^{b}(t)$ to find its maximum:

$$\frac{\partial L}{\partial w_{i,j}^{b}(t)} = (-V)r_{i,j}^{b}(t) - r_{i,j}^{b}(t)[U_s(t) + L_i(t)]. \tag{3.28}$$

The derivative used to obtain the dynamic resource allocation algorithm, where for each PRB, the UE with a minimum value of $(-V)r_{i,j}^{b}(t) - r_{i,j}^{b}(t)[U_s(t) + L_i(t)]$ is selected to be served. Finally, the slice manager updates the virtual queues and the same procedure occurs in the next slot.

3.3.3 Performance Evaluation

In this subsection, we demonstrate the main findings of our model. Given that slice isolation is the main focus, initially we demonstrate why a radio resource-based solution is not sufficient

Table 3.4: Performance evaluation parameters

Parameter	**Value**
Number of Slices	3
Number of BSs	$\{2, 3, 4, 5\}$
Number of users per slice	$\{4, 5, 6, 7\}$

for frequency selective wireless channels and highlight the importance of considering a QoS-based isolation definition for network slices.

As mentioned, the preservation of slice isolation becomes even more complex when a multi-BS scenario is considered. Moreover, when the network dimensions increase not only in terms of BSs, but also in terms of UEs, the ability to preserve isolation is even more challenging. In that regard, in this section we show the performance of the Lyapunov optimization while increasing network dimensions, specifically with respect to UEs and BSs. Given that the Lyapunov optimization is an approximation technique, there is a trade-off between optimality and convergence time of the algorithm. In the remainder of this chapter, we will refer to convergence time as the time instance within the simulation, where all the Lyapunov queues are stabilized, indicating a satisfaction of the QoS requirements. As detailed previously, the Lyapunov parameter V denotes this trade-off. Additionally, we provide results with respect to V, which gives a hint about the fine-tuning of this parameter for a specific system setup. Finally, we compare the Lyapunov approach with alternative state-of-the-art solutions in terms of throughput maximization and ability to satisfy UE requirements. We do this to demonstrate the effectiveness of our approach. For our simulation we consider the overall parameters presented in Section 3.1.1, which are shown in Table 3.2, whereas particular performance evaluation parameters are given in Table 3.4.

3.3.3.1 Isolation Comparison

The initial results aim at identifying the importance of considering a QoS-based slice criteria compared to a radio resource-based one. For that purpose, we compare our algorithm with one of the most relevant approaches in the state of the art with respect to radio resource-based solutions [DOr+19]. In [DOr+19], the authors introduce the concept of linked PRBs, meaning that identical PRBs should be allocated to the same slice within multi-BSs in order to avoid interference with the use of sophisticated power management techniques. In that regard, they propose a heuristic approach, called Most Linked First (MLF), which starts resource allocation with those slices that have the most linked resources. While the inter-slice allocation problem is solved with MLF, intra-slice scheduling (i.e., how the resources are distributed to UEs within a slice) in [DOr+19] is not explicitly mentioned. For a fair

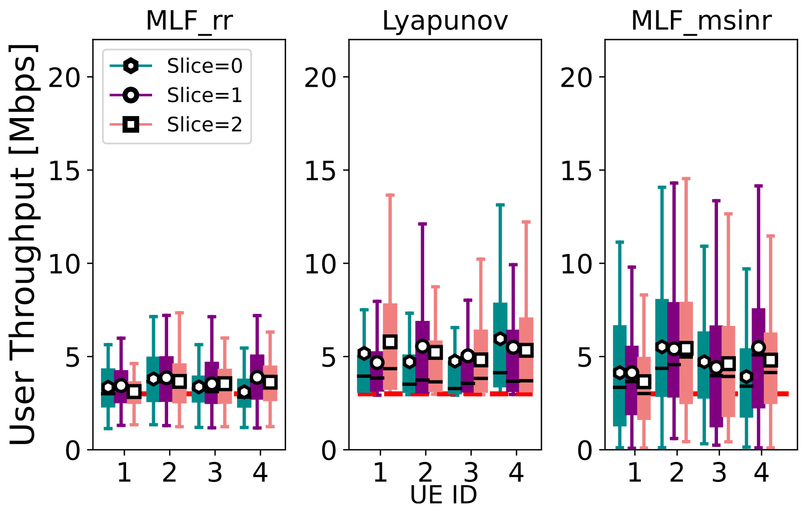


Figure 3.9: Isolation comparison for MLF_rr, MLF_msinr and Lyapunov approach for 3 Mbps minimum UE requirement in a 2 BS scenario. While the Lyapunov approach can effectively satisfy all UEs, MLF_rr and MLF_msinr cannot satisfy all UEs in all runs.

comparison, we define two approaches, namely MLF Round-Robin (RR) (MLF_rr) and MLF maximum SINR (MLF_msinr), where the second algorithm defines the way how resources are distributed within a slice.

Fig. 3.9 demonstrates the difference between the Lyapunov optimization and RAN slicing enforcement MLF [DOr+19] in a frequency selective wireless environment. The x-axis represents a UE within a slice and the y-axis denotes the achieved throughput for each UE. In total, there are 12 UEs distributed equally over 3 slices served by 2 BSs. For the MLF approach, a minimum requirement per slice is assumed. Given that all slice requirements are identical, the number of resources is equally distributed among slices. In turn, for the Lyapunov approach representing a per-UE-based scheme, a minimum requirement (3 Mbps per UE) is selected based on a challenging, but feasible scenario, generated from simulation results. The simulations are performed for 50 runs, where each run consists of 5000 time slots with time and frequency variations every 20 ms (i.e., the coherence time of the wireless channel). The boxplots for all runs are shown in Fig. 3.9. The results illustrate that for MLF_rr and MLF_msinr none of the UEs achieves the minimum QoS requirements in all runs. Although a minimum requirement in terms of resources is provided to slices, neglecting the UE channel conditions when performing the resource allocation leads to throughput degradation, which can be avoided by assigning carefully the most suitable channel per UE. Alternatively, the Lyapunov optimization can serve all UEs while satisfying their minimum QoS requirements for all runs. We can therefore conclude that in a frequency selective wireless environment a radio resource-based solution cannot guarantee a minimum QoS requirement for all UEs, thus a QoS based approach is mandatory for the RAN slicing problem to provide isolation.

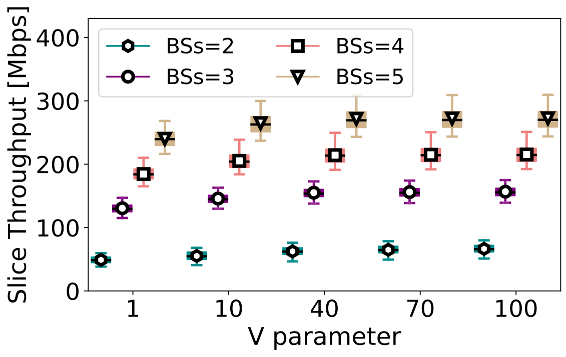


Figure 3.10: Aggregated slice throughput for the Lyapunov optimization with increasing number of BSs for 12 UEs and 3 slices for various V shape parameters. The overall slice throughput increases with increasing V and the number of BSs.

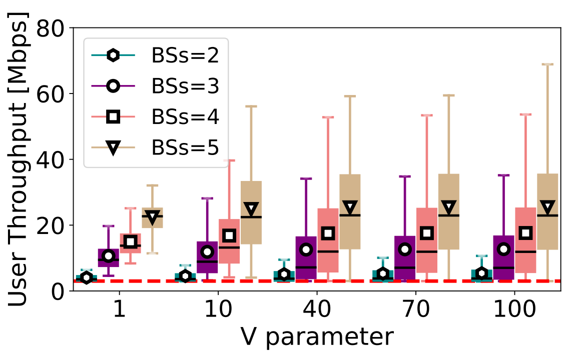


Figure 3.11: Individual UE throughput for the Lyapunov optimization with increasing number of BSs for 12 UEs and 3 slices for various V shape parameters with a minimum requirement of 3 Mbps per UE. The Lyapunov optimization can effectively preserve a minimum throughput for all UEs.

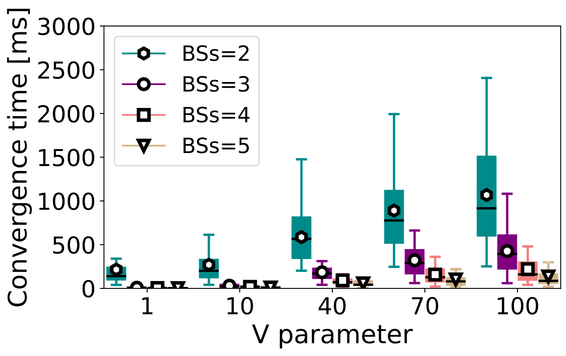


Figure 3.12: Convergence time for the Lyapunov optimization with increasing number of BSs for 12 UEs and 3 slices for various V shape parameters. The convergence time of the algorithm increases with increasing V, but decreases with the number of BSs due to higher achievable throughput.

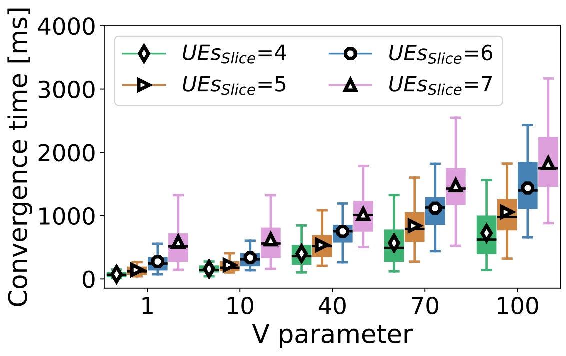


Figure 3.13: Convergence time for the Lyapunov optimization with increasing number of UEs for 2 BSs and 3 slices for various V shape parameters. The convergence time of the algorithm increases with increasing V and with the number of UEs due to the strict isolation constraints.

3.3.3.2 Impact of Number of BSs

Further investigations are conducted for the proposed Lyapunov optimization with respect to the number of BSs in the network. In principle, when the number of BSs increases in the network and given the limited amount of frequency bands, the probability of BSs interfering with each other also increases. The interference becomes even more noticeable while considering a small BS scenario like inside an aircraft cabin, where BSs are placed closer to each other, thus producing higher interference. Therefore, the preservation of isolation becomes more challenging, leading to the need to consider carefully a RAN slicing algorithm.

In order to demonstrate the effectiveness of our approach, we illustrate the outcome of adding BSs in the network with respect to the throughput of network slices and individual UEs, while keeping the number of UEs fixed to 12 over all slices. The results are depicted in Fig. 3.10 and Fig. 3.11, respectively. The x-axis represents the V shape parameter of the Lyapunov optimization. As already mentioned, this parameter depicts the trade-off between optimality (i.e., increasing slices' throughput and queue stability), which in our case directly translates to the ability to satisfy the constraints (i.e., preserve the minimum throughput requirement per each UE). The y-axis shows the total slice throughput in Mbps for Fig. 3.10 and the individual UE throughput in Mbps for Fig. 3.11. The simulations have been conducted over 50 runs and the boxplots are drawn to show the values of the quartiles of the results. Given that all slices are identical, we combine all the UEs of slices into single boxplots to illustrate the results of 50 runs. For the simulations, the minimum requirement per each UE within the slice has been fixed to 3 Mbps, whereas the slice arrival $\bar{\lambda}_s$ is also 3 Mbps on average. As observed from Fig. 3.10, the overall slice throughput increases with increasing the V parameter. Moreover, by increasing the number of BSs, the slice throughput increases as well. The individual UE throughput varies on average between 4 and 6 Mbps for 2 BSs, when V ranges from 1 to 100. For 5 BSs, the individual UE throughput varies between 22 and 26 Mbps on average, with V ranging from 1 to 100.

As mentioned above, while interference increases with the increase of BSs, the distance to the UEs of each slice decreases. In that regard, a careful allocation of PRBs demonstrates that the interference problem can be omitted and the diversity gains are much higher. Furthermore, Fig. 3.11 illustrates that irrespective of the V shape parameter each UE's individual throughput is achieved, demonstrating the effectiveness of preserving the slice isolation when using the Lyapunov optimization.

Finally, to portray the trade-off between optimality and queue stability, we show the convergence time of the algorithm with respect to the V shape parameter in Fig. 3.12. The x-axis represents the V shape parameter, whereas the y-axis the convergence time in ms. As it can be seen, the convergence time increases with increasing V for all the scenarios. However, it decreases with increasing the number of BSs. For 2 BSs the convergence time of a single run consisting of 5000 slots, on average varies from 200 up to 1050 ms, with ranging V from 1 to 100. In other words, it takes approximately between 200 and 1050 slots (with V in the range 1 to 100) for the algorithm to obtain the queue stability and to satisfy QoS requirements. Alternatively, for 5 BSs the average varies between 7 and 120 ms. The intuition behind this result is that increasing the number of BSs increases the potential throughput when a careful RAN slicing algorithm is applied and as such the individual UEs requirements are achieved faster.

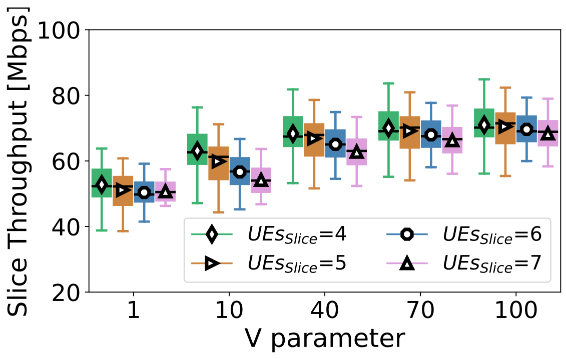


Figure 3.14: Aggregated slice throughput for the Lyapunov optimization with increasing number of UEs for 2 BSs and 3 slices for various V shape parameters. The overall slice throughput increases with increasing V and decreases with the number of UEs as it is harder to fulfill the user requirements.

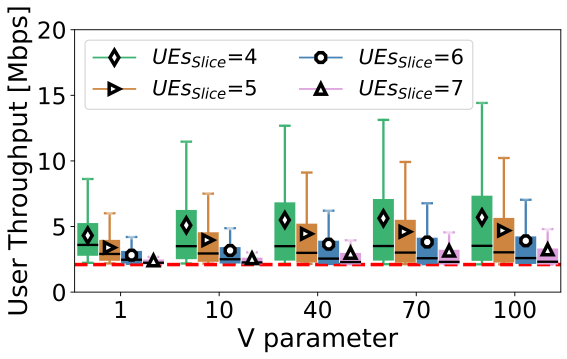


Figure 3.15: Individual UE throughput for the Lyapunov optimization with increasing number of UEs for 2 BSs and 3 slices for various V shape parameters with a minimum requirement of 2.2 Mbps per UE. The Lyapunov optimization can effectively preserve a minimum throughput across all UEs.

3.3.3.3 Impact of Number of UEs

Similarly to the effect of the number of BSs to our algorithm, in this part we demonstrate the impact of UEs in the overall performance. By keeping the number of slices fixed to 3 and the number of BSs to 2, we show results with increasing number of UEs within a slice. Initially, we demonstrate the overall slice throughput, slice isolation and finally, the convergence time.

Fig. 3.14 demonstrates the overall slice throughput with respect to the number of UEs within a slice, whereas Fig. 3.15 the individual throughput achieved by each UE within a slice. The x-axis represents the V shape parameter, while the y-axis the overall slice throughput. Different boxplots are drawn for various UE sizes and the distribution for 50 runs is illustrated. It is worth noting that in this scenario for increasing number of UEs for 2 BSs the minimum requirement has been set to 2.2 Mbps per UE, otherwise the system is unable to provide the throughput due to the solution infeasibility. Again, all UEs have been combined together and the results are demonstrated in boxplots. In Fig. 3.14 we can observe that increasing the V shape parameter increases the overall slice throughput for all considered scenarios. The throughput ranges from 55 to 70 Mbps, on average, for 4 UEs per slice and 50 to 68 Mbps for 7 UEs per slice, when considering a V shaper parameter from 1 to 100. It is observed that increasing the number of UEs does not bring a big benefit in the overall slice performance. The rationale behind this result is greatly related to the UE's minimum requirement that has to be preserved for each slice, which becomes challenging when more UEs exist in the system, especially since UEs are randomly selected from the aircraft plan in [Boe] and experience distinct channel variations. Nonetheless, Fig. 3.15 shows that for each V shape parameter the minimum requirement is achieved, demonstrating the slice isolation.

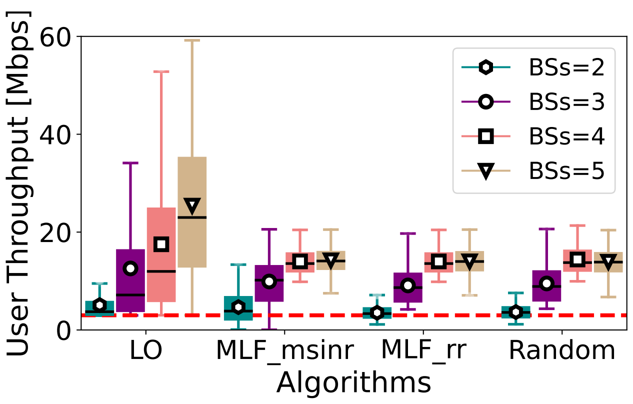
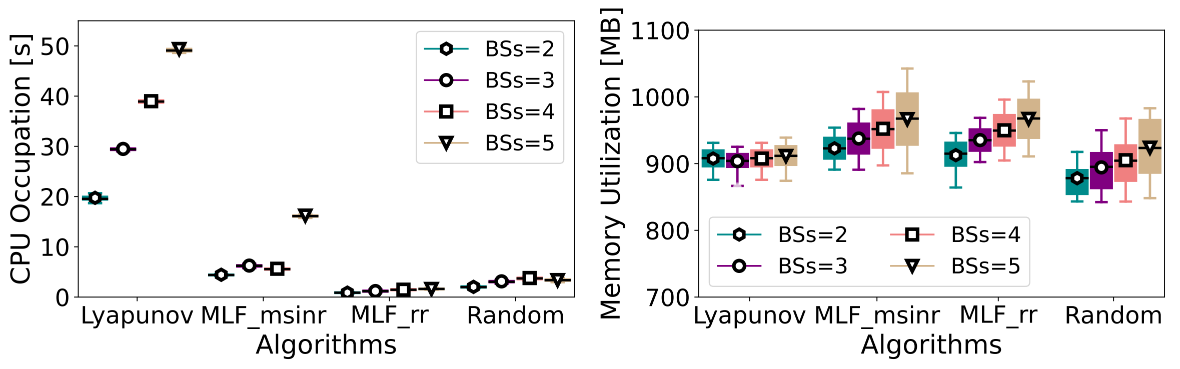


Figure 3.16: Comparison of the Lyapunov optimization with state-of-the-art solutions for varying number of BSs for 3 slices and 12 UEs, with a minimum requirement of 3 Mbps. The Lyapunov optimization can double the throughput compared to the best alternative solution MLF_msinr for 5 BSs. Moreover, it is the only technique that can guarantee all UEs' minimum requirement in all runs.

Finally, we illustrate the trade-off between the optimality and convergence time of our proposed solution with respect to UEs in Fig. 3.13. The x-axis represents the V shape parameter, whereas the y-axis depicts the convergence time in ms. As it is portrayed in the figure, the convergence time increases with increasing V, as expected. Moreover, we notice that while the number of UEs increases the convergence time also increases. Considering 4 UEs per slice, the convergence time increases from 60 ms for $V = 1$ to 700 ms for $V = 100$, on average. Alternatively, for 7 UEs per slice, a variation between 600 ms for $V = 1$ and 1800 ms for $V = 100$ is experienced on average for the convergence time. This behavior can be explained with the ability to preserve each UE's minimum requirement, which is harder when the number of UEs increases in the network. However, as demonstrated above, the Lyapunov optimization is still able to achieve it. Again to be stressed, that the algorithm is used to solve the problem for a total of 5000 slots, thus the convergence time still remains within reasonable values.

3.3.3.4 Comparison of Algorithms: Throughput Maximization

Here, we demonstrate a comparison among two MLF [DOr+19] variants, our optimization approach as well as a random solution, where resources are randomly and uniformly distributed over slices and UEs within a slice. Differently from Fig. 3.9, where we demonstrated the isolation comparison among Lyapunov and MLF variants only for a 2 BS scenario, here we investigate the difference also for higher number of BSs (i.e., up to 5). We fix the number of UEs to 4 per each slice and we consider 3 slices. We set up the minimum requirement for our approach to 3 Mbps per UE and $V = 40$, which demonstrates a good trade-off between optimality and convergence time. Alternatively, for the other approaches we consider a radio



(a) CPU utilization (b) Memory utilization

Figure 3.17: Comparison of the Lyapunov optimization with state-of-the-art solutions for varying number of BSs for 3 slices and 12 UEs with minimum requirement of 3 Mbps for $V = 40$ with respect to CPU and memory utilization. While the overhead for the memory utilization for the Lyapunov approach is minimal, the CPU consumption induces a higher overhead, but it is tolerable as compared to the state of the art.

resource-based solution, where each slice requests equal resources. The comparisons are illustrated in Fig. 3.16, where the x-axis represents the different algorithms, whereas the y-axis the individual UE throughput. The results are demonstrated for various BSs deployed in the network for 50 runs, where all UEs among slices are combined in single boxplots.

As can be observed from Fig. 3.16, the Lyapunov optimization approach, proposed in this chapter, achieves slice isolation for all UEs compared to 50% of the best state-of-the-art solution (i.e., MLF maximum SINR). Moreover, it further achieves better overall performance with increasing BSs. Particularly, for 5 BSs the throughput is almost doubled on average, 26 Mbps compared with 14 Mbps of the best state-of-the-art solution. Again, these results demonstrate the importance of considering a QoS based solution for frequency selective channels and indicate the improvements in network performance (i.e., throughput that can be accomplished with the Lyapunov approach).

3.3.3.5 Comparison of Algorithms: Overhead

In this part of the evaluation, we elaborate more on the computational cost overhead of the proposed Lyapunov optimization approach compared to the state of the art. Similarly to the previous comparison, there are in total 3 slices with 4 UEs each. The minimum requirement for our approach is set to 3 Mbps per UE with a V parameter of 40, which presents a good trade-off between optimality and convergence time. Results demonstrating the CPU and memory utilization of each algorithm for various BSs deployed in the network for 50 runs are illustrated in Fig. 3.17a and Fig. 3.17b, respectively. Fig. 3.17a shows the CPU occupation of each algorithm in seconds. According to the figure, the CPU utilization increases with increasing number of BSs. The Lyapunov optimization records the highest occupation time with up to 50 s for 5 BSs compared to the maximum SINR MLF solution with 18 s, demonstrating an induced overhead with respect to CPU consumption. Nonetheless, an improved code parallelization of the Lyapunov solution can further reduce the CPU overhead. Furthermore, we stress that the CPU consumption does not reflect the convergence time of the algorithm as shown in the previous results. Fig. 3.17b, illustrates a similar increasing trend of the memory utilization with increasing the number of BSs for all the algorithms. However, the Lyapunov optimization records a smaller memory utilization on average apart from the random solution, concluding that the overhead of the Lyapunov optimization with respect to memory is minimal.

3.4 Practical Implementation

Up to now, we focused on proposing a solution that can be adopted for standardization purposes and which could be practically realized. To that end, we envision the presented architecture in Fig. 3.1 to be incorporated into existing SD-RAN open-source platforms such as `FlexRAN` [Fou+16] combined with OpenAirInterface [Nik+14]. `FlexRAN` is designed such that it allows the communication with the application layer (i.e., slice owners) utilizing APIs similarly to our approach. Moreover, `FlexRAN` provides a protocol that allows the communication with the underlying BSs and enables the possibility to alter and control the resource allocation process. Likewise, OpenAirInterface is a platform for 5G experimentation that has full compatibility with `FlexRAN`. The OpenAirInterface BS implementation provides statistics about the UEs and the channel variations as well as the throughput achieved with each PRB. That entails that this information can be utilized by slice manager's statistics block in Fig. 3.1 to perform the optimization. Finally, the resource allocation is envisioned to operate within the `FlexRAN` controller in order to allow flexibility in the decision making with respect to the slice management. Although the integration of the proposed algorithm is not yet implemented in `FlexRAN` or OpenAirInterface, it remains an interesting area of research area to relate he simulation results with a real implementation.

3.5 Controlling Next-Generation Software-Defined Radio Access Networks: A 5G Standardization Perspective

As mentioned previously in this thesis, softwarization and programmability are the pillars for provisioning the adaptability and flexibility in 5G networks. Generally speaking, the control and data plane separation provided by SDN is a perfect example of a technology that can fulfill the 5G requirements. However, while the 5GC and NR functions are well defined in 3GPP standardization [3GPb; 3GPd], novel concepts such as SD-RAN still remain unspecified in the standard.

In this chapter, we have demonstrated so far the benefits of SD-RAN from the mathematical point of view, while identifying potential benefits for the RAN architecture. But, we have not hinted the opportunities of a possible integration of the developed algorithms into the 3GPP architecture. Thus, in this section we demystify the necessary tools for RAN slicing integration into the 5G architecture while introducing Radio Access Network Control Function (RANCF). RANCF corresponds to a realization of an SD-RAN controller in a 5G NR architecture. In principle, RANCF is envisioned for the control and coordination of multiple gNBs (i.e., BSs in 5G terminology). In compliance with the 5G system architecture, RANCF is defined as a flexible and easy to integrate network function. From the network operators' and vertical stakeholders' perspective, RANCF alleviates the challenge of deploying and managing network slices by enabling a coordination mechanism between the NR and 5GC, which is missing in the existing 5G architecture. Moreover, while controlling multiple gNBs, RANCF maintains a centralized network view that can increase radio resource efficiency with potential lower signaling overhead compared to distributed approaches. To enable this support, we define interfaces and protocols that can be used for the interaction of RANCF with existing 5G functions and shed light on its control functionalities. Finally, we leverage network slicing as the main 5G use case and show how RANCF acts for network slice selection, life cycle and management.

3.5.1 Radio Access Network Control Function Overview

The overview and connection of RANCF with the NR and 5GC is depicted in Fig. 3.18. In the latter, we elaborate in detail all the connections.

3.5.1.1 RANCF Interaction with NR Functions

The application of the SDN concept to the 5G architecture manifests in the decoupling of control plane and user plane functions. While this principle is well established in the 5GC, in NR, control plane and user plane functions are mainly residing in the gNB. With network slicing arising as one of the main features of 5G, there have been vast amount of proposals to enable the control of network slices in the RAN via SD-RAN controllers, where an inter-slice scheduler (i.e., SD-RAN controller) steers the resource allocation over multiple intra-slice schedulers, which are in charge of scheduling decisions within a network slice [Pap+19b; CNS18].

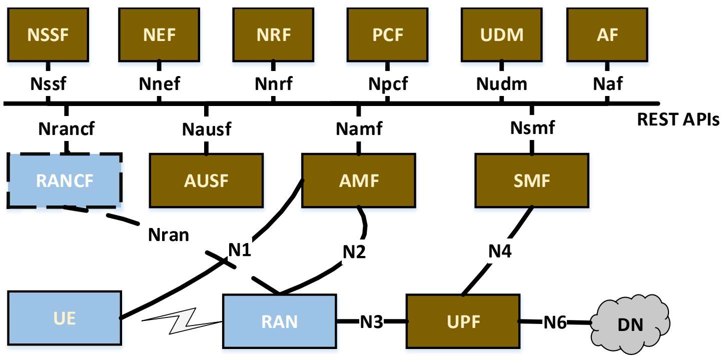


Figure 3.18: Envisioned 5G functional architecture based on the 3GPP Release 15, including the proposed RANCF function. NR components are marked in blue and 5GC functions in brown.

In the context of providing an SD-RAN controller within the 5G architecture, we propose the RANCF. RANCF is mainly responsible for the control of the NR handling decisions such as scheduling, handover, power management. Therefore, in our envisioned 5G architecture portrayed in Fig. 3.18, RANCF is highlighted as a new 5G control function. Moreover, we define the **Nran** interface needed for the interaction of RANCF with the underlying RAN infrastructure as well as **Nrancf**, which is needed for RANCF's interaction with other existing 5GC functions (i.e., AMF, SMF, AF). A detailed description of the aforementioned functionalities, protocols and interfaces was provided in Section 2.1.2.

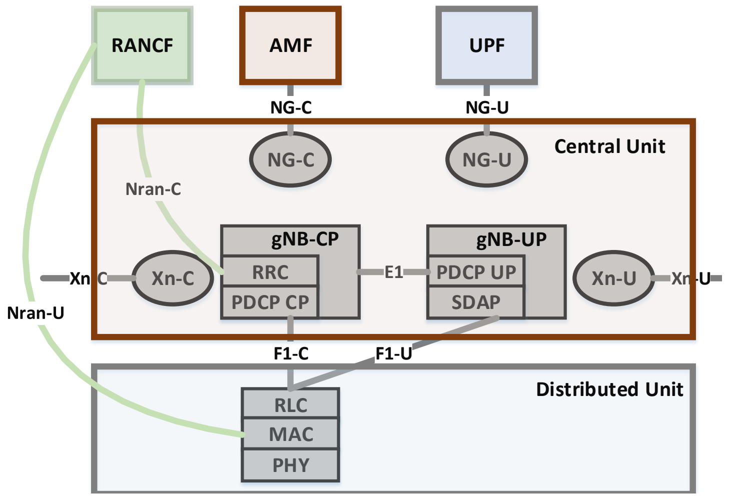


Figure 3.19: Envisioned NR functional architecture based on a separation of functions between distributed units, centralized units and an SD-RAN controller (RANCF). NG-C and NG-U are interfaces to the 5GC. The proposed interfaces to the envisioned RANCF are Nran-C and Nran-U.

3GPP proposes a functional split among the functions within a gNB presented in Section 2.1.3.1. Whereas several different RAN function splits are possible, we focus on the RLC-PDCP function split, proposed in [3GPd], as a typical example. However, any functional split can be supported by RANCF. For the RLC-PDCP function split, in the DU (i.e., gNB-DU), functions like RLC, MAC and PHY are located, whereas PDCP and RRC functions are located in the so called Central Unit (CU)s (i.e., gNB-CU). Within the CU, there is a logical separation among the CP and the user plane of the gNB, referred to as gNB-CU-CP for the control plane and gNB-CU-UP for the user plane, respectively. The connection between the gNB-CU and gNB-DU is realized by the F1 interface, which in turn is separated in F1-C for the control plane and F1-U for the user plane. Likewise, the connection between the gNB-CP and user plane is realized by the E1 interface. In a similar fashion, the RAN control and user plane are connected to the 5GC control and user plane by utilizing the **NG-C** and **NG-U**, accordingly. Finally, the RAN domain can contain multiple gNBs which are interconnected via the **Xn-C** and **Xn-U** interfaces.

In the aforementioned 5G architecture, each gNB is responsible for the control of its user plane. There exists a communication among gNBs, however the main purpose is to facilitate the handover procedure. Such a distributed approach might lead to a suboptimal solution and furthermore it may increase the signaling overhead if additional tasks, such as network slice scheduling and power management, are added. In contrast, in our proposed architecture, the control over multiple gNBs is centralized in RANCF. In that regard, due to a broader view of the network, RANCF can make smarter decisions and therefore boost the network performance, but also potentially reduce signaling overhead since only the affected gNBs will be invoked by RANCF.

Following the logic of this architecture, we define the interfaces of RANCF with the appropriate functions in the gNB and illustrate our envisioned concept in Fig. 3.19. As described, the main functionality of the RANCF is to provide a centralized control of the RAN domain realizing an SD-RAN controller. This control for example may influence scheduling decisions, power allocation, modulation and coding schemes of specific gNBs. The gNB function in charge of the aforementioned tasks is the MAC. Hence, we believe that a control connection between the RANCF and the MAC layer of the distributed unit of the gNB should be established. This connection is depicted in Fig. 3.19 as **Nran-U**. Similarly, RANCF can be utilized to coordinate decisions of the control plane of the gNB, which is located in the CU. These control decisions may involve, but are not limited to, altering RRC decisions on the QoS management, handover decisions, establishment and modification of SRBs and DRBs. Hence, in our proposed concept we envision a control communication between RANCF and RRC, namely **Nran-C**. Both **Nran-U** and **Nran-C**, denoted by **Nran** in Fig. 3.18, can be for instance protocols utilized by SD-RAN controllers similar to `FlexRAN` [Fou+16] and `5G-EmPOWER` [CKR19].

3.5.1.2 RANCF Interaction with 5GC Functions

While in the core part of the 5G architecture, SMF is responsible for the control decisions, for instance selecting the appropriate UPF for network slices, in the RAN part of the 5G architecture, the RANCF will be responsible for such tasks. It is therefore crucial that SMF and RANCF share a view of the network with each other in order to provide end-to-end guarantees. As illustrated in Fig. 3.18, the control information between SMF and RANCF can be exchanged by using REST-like protocols. Moreover, RANCF should maintain communication with other control functionalities of 5GC, e.g., with AMF to gather information about UEs' mobility. REST APIs can be used here as well. Furthermore, RANCF can support diverse application requirements in RAN. Specific requirements may even imply changes of the scheduling decisions or handovers. In such cases, the AF can utilize the **Nran** interface and apply application-specific policies directly to RANCF using REST API and in turn, RANCF should translate these policies to a format that is understood within the RAN.

3.5.2 Network Slicing in 5G Supported by RANCF

According to the 5G information models [3GPc], the implementation of a network slice is referred to as Network Slice Instance (NSI). An NSI can consist of multiple Network Slice Subnet Instance (NSSI)s. Namely, an NR NSSI and a 5GC NSSI. Each of the created NSSIs contains multiple NFs, which are required to meet the QoS of the NSI. In the following, we elaborate more on the concept of network slicing with respect to the state of the art and in 3GPP standardization. In particular, we use the example of network slicing to illustrate the role of RANCF in the 5G architecture.

3.5.2.1 End-to-end Network Slicing

5G applications require stricter isolation guarantees due to stringent delay requirements. The question that arises is how such guarantees can be realized.

State-of-the-art proposals answer this question by introducing possible solutions for end-to-end network slicing. For example, the authors in [CR19] propose a flow-based network slicing approach, where both RAN and core are controlled by an SDN-like controller. The communication between the RAN and the core has to surpass an Open vSwitch. Slices are created by the SD-RAN controller. Additionally, the slice information is sent to the core network controller which in turn deploys the appropriate flow rules on the Open vSwitch. On an other note, the work in [NVH18] introduces an end-to-end slicing approach to enable low-latency edge applications in the network. It also utilizes Open vSwitch to steer the information towards an edge computation platform to cater for delay reductions for specific delay sensitive

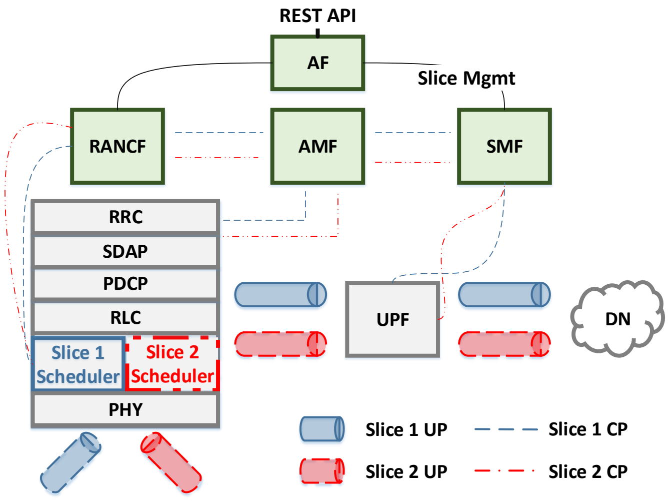


Figure 3.20: Illustration of the interaction of 5G functions and RANCF to support end-to-end network slicing. Dashed lines show control information flows. Tube shapes represent isolated user plane slices from the UE through the 5G RAN and core to the DN.

applications. Both proposals provide viable solutions for coordinating network slices in the core as well as in the RAN part. Yet, a concrete mapping and embedding of such solutions to the 5G architecture standard is not explicitly shown in either of them.

Driven by existing solutions in the literature, we depict our view of the end-to-end network slicing concept in Fig. 3.20. We argue that a coordination among the NR NSSI and 5GC NSSI is necessary to enhance network performance, since a lack of coordination cannot achieve the required level of isolation of the NSI. For instance, if the operator supports network slicing only in the RAN, there is no guarantee that the QoS can be preserved if the appropriate UPFs are not reserved in the core. Likewise, if only the core network is slice-aware, there is no guarantee that the required amount of radio resources associated with each network slice can be granted. For each deployed network slice, apart from time and frequency resources in the RAN, the appropriate backhaul bandwidth should be reserved in the tunnel between the RAN and UPF. Hence, the control coordination among the RAN and core becomes mandatory.

In the following, we map the SD-RAN concept manifested in the proposed RANCF in analogy with the already developed open-source platforms [CR19; NVH18] for an end-to-end slicing solution to the current 5G functional architecture. SMF can be understood as an SDN controller of the core network, whereas RANCF resembles the SD-RAN controller. An east/west communication API is therefore needed for control and coordination. The REST API envisioned for the communication amongst the functions in the 5GC, presented in Fig. 3.18, can be used for that purpose. In that regard, a protocol is needed for control information exchange. The protocol shall support the 5G information model known as Network Resource Model (NRM), based on Extensible Markup Language (XML) or Javascript Object Notation

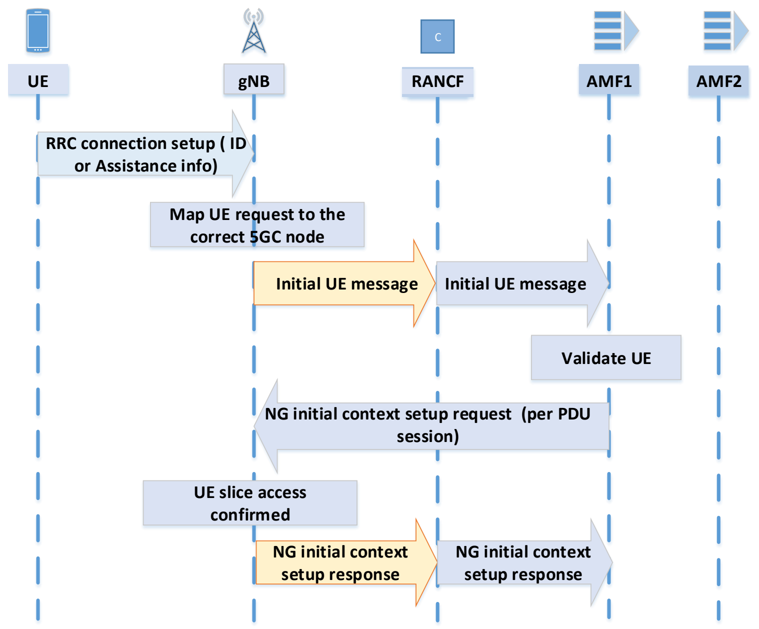

UE
gNB
RANCF
AMF1
AMF2
RRC connection setup (ID or Assistance info)
Map UE request to the correct 5GC node
Initial UE message
Initial UE message
Validate UE
NG initial context setup request (per PDU session)
UE slice access confirmed
NG initial context setup response
NG initial context setup response

Figure 3.21: Network slice selection signaling in 5G, including RANCF.

(JSON) formats. Using such data models, the information between the NR NSSI and 5GC NSSI can be exchanged with respect to slice resource utilization. In this way an end-to-end flow control of the NSI can be maintained.

3.5.2.2 Network Slice Creation and Management

The creation and management of network slices raises many questions and remains an important topic for research. In the following, we address the open questions regarding network slice creation/management using our proposed RANCF and based on recent 3GPP studies [3GPc]. From the logical point of view, a network slice is requested by a service or a network operator. Therefore, the slice creation is offloaded to the AF in 5G. We believe that the AF will be in charge of the network slice management coordination process and slice acceptance admission. It should be able to gather information from both the core and RAN part of the 5G network to sustain a global view and act depending on the availability of both SMF and RANCF. In turn, decisions on the amount of UEs in a slice are then part of the admission control of individual network slices. In our proposed 5G architecture, the deployment part of the NSI is thus a task of the application functions. As illustrated in Fig. 3.20, the service can be part of the same operator or requested by another operator or service provider out of the operator's domain by utilizing a REST API.

3.5.2.3 Network Slice Selection

While network slice creation and management is still unclear, 3GPP in Release 15 [3GPa] has already standardized the slice selection procedure. Initially, each AMF connected to the RAN node is requested to identify the list of slices that they can support. This request is referred to as Next Generation (NG) setup request. Each UE can then select the preferred network slice to attach to by including an identification or an assistance information. This information is then utilized by the RAN in order to steer the UE request to the appropriate AMF. The AMF then validates the UE by requesting information from the NSSF, whether the UE is allowed to attach to the requested slice. If the UE is successfully attached, SMF selects the appropriate UPF for the UE and initiates a Protocol Data Unit (PDU) session. The RAN is then notified by receiving a NG context setup request. It is then the RAN's duty to perform resource allocation for a successful transmission to be initialized. Since the RANCF is responsible of scheduling decisions for network slices, it should be notified about all the slices that are created, as well as the users of each network slice. This means that all the slice initiation/termination information of users should be noted by the RANCF in order to have a clear view of the network. Thus, in our view the 3GPP slice selection procedure should involve RANCF, as depicted in Fig. 3.21.

3.5.3 RANCF Implementation Details

We have emphasized the importance of RANCF in a 5G architecture and we have highlighted its interaction with existing 5GC and RAN functions to fulfill the network slicing concept, in particular, proposing concrete protocols and interfaces. Moreover, we showed the importance of RANCF from the network operators' and verticals' perspective that lies in relieving the management and orchestration burdens of a network slice. In light of the benefits that RANCF brings to 5G, the potential overhead has to also be taken into account, mainly from additional introduced signaling overhead. Intuitively, the signaling overhead is affected by the number of slices, users of a slice and gNBs that the RANCF is controlling as well as the size and frequency of control message updates.

In order to compare RANCF with other proposed solutions or to evaluate its efficiency we can utilize Key Performance Indicator (KPI)s that consist of the life-cycle of a slice, such as *slice deployment time*, *slice scalability* and *slice reconfiguration time* as elaborated in [KT19]. Furthermore, slice efficiency KPIs extend also to radio (e.g., bandwidth) and mobile core (e.g., backhaul bandwidth, CPU cores, memory) resource utilization.

From a standard's perspective, RANCF could easily be integrated into the 5G architecture. That means there are no particular required changes to the 5G architecture. Since with RANCF we follow the logic of a service-oriented architecture as proposed in 5G, we introduce our envisioned RANCF as a new function. We utilize the existing REST protocol proposal for the envisioned required communication. Hence, we infer that our function can be easily adopted by the 5G architecture with minimum extra effort.

3.6 Summary

In this chapter, we provided a system model for RAN slicing based on the concepts of *programmability* and *virtualization*, which originate from the SD-RAN paradigm. Initially, we considered a single-BS scenario, for two types of RAN slices, the first which is delay-critical and the second which is throughput-critical. Both types were sharing resources in a shared infrastructure. We designed an approach based on the Lyapunov optimization that minimizes resource consumption while satisfying slices' requirements and simultaneously maintaining slice isolation.

Alternatively, considering the multi-BS scenario, where we need to take the interference into account, we presented a clear RAN slicing definition up to the UE level. Leveraging wireless channel characteristics, we dynamically re-assign wireless resources, which helps improving the system throughput compared to state-of-the-art approaches and at the same time guarantees UE isolation within RAN slices.

Initial implementation details of the proposed approach as well as a proposal for the integration of the developed system in the 5G architecture were highlighted in this chapter. In that context, we demonstrated how SD-RAN can be provided as a 5G function in the process of improving the RAN slicing efficiency.

In order to provide UE guarantees in SD-RAN environments, apart from the optimization methods shown in this chapter, performance guarantees have to be maintained in other system components such as SD-RAN controllers. Additionally, the overhead of RANCF as well as of SD-RAN's controllers has to be quantified in order to verify their applicability in 5G scenarios and different use cases. In that regard, in the next chapters of this thesis, we focus in particular on demystifying important performance aspects regarding the SD-RAN implementation.

Chapter 4

Performance Analysis of Software-Defined Radio Access Network Platforms

The previous chapter of this thesis has demonstrated that the SD-RAN paradigm boosts RAN performance and facilitates RAN slicing. However, since SD-RAN controllers become a vital component of the envisioned infrastructure, they also pose significant threats to the smooth operation of the overall network. For instance, let us assume a centralized SD-RAN setup where the entire control plane of RAN is controlled by a single SD-RAN controller. In such a case, the latter can potentially become a single point of failure. The failures do not occur only due to the internal behavior of SD-RAN controllers, but also due to the high network load that is imposed from the data plane, in which BSs and UEs operate. Therefore, a deeper understanding of the actual behavior of SD-RAN controllers in the control plane, especially when they are forced to operate at their capacity limits, becomes of utmost importance. Hence, in this chapter, we focus explicitly on the modeling and analysis of state-of-the-art SD-RAN controllers and we further provide design guidelines for the internal working behavior of the existing solutions.

In order to support 5G/6G heterogeneous applications, like tele-operations, online gaming and IoT with thousands of connected devices [Oug+19], research as well as 3GPP standardization [3GPa] suggest a full softwarization of the mobile network by means of SDN. On the one hand, SDN alleviates the deployment of network slicing [All16], which leads to the possibility to accommodate and maintain large networks with thousands of devices. On the other hand, SDN is also intended to boost the network performance by flexibly deploying functionalities closer to the UEs. Such applications include Mobile Edge Computing (MEC), which can cater for the stringent delay requirements of 5G/6G applications.

Recently, the concepts of network *programmability* and *virtualization* have also been embraced by the industry. As it is reported by Rakuten [Rak20], the programmability provided by SDN can lead to reliable network deployments. Additionally, in the context of next-generation RANs, the O-RAN alliance [O-Ra] is investing in open, intelligent and virtualized systems, demonstrating the importance of softwarization in the 5G/6G era.

However, while softwarization has been mainly applied in data centers and wired networks [Jin+13; CYY16], recently the attention started to shift towards the RAN. In particular, RAN can benefit from the centralization concept of SDN in order to improve spectrum efficiency, power management, interference cancellation and handovers [Fou+16; CKR19; Pap+20b], which require tight cooperation among BSs. Hence, the SDN concept has been adopted by the RAN as lately shown in standardization [3GPd], where a clear separation of the control and data plane is proposed. In the heart of the SD-RAN architecture lies the SD-RAN controller, where the control over the whole network is centralized. Intuitively, the SD-RAN controller becomes a potential bottleneck given the high signaling overhead imposed by thousands of devices in the network. While vast ongoing research is based on concepts containing SD-RAN controllers [Cos+18; NVH18; CR19; Ram+18; Kou+19; Gar+18b; Kok+11; SCB19; Gar+18a; Aya+19], only a few performance evaluations are conducted [Fou+16; CKR19] for small network dimensions up to 50 devices.

The main research question we address in this chapter is whether current SD-RAN controllers can sustain the scale of future networks [Oug+19], such as IoT (e.g., $>$ 5000 connected devices). We identify the lack of a scalable performance measurement tool as a problem since the performance guarantees in a realistic deployment remain unknown and the possibility for commercial use is questionable. However, designing a tool that is able to scale to realistic 5G/6G network dimensions is a non-trivial task. On the one hand, current open-source SD-RAN controllers are implementation-specific; they are designed for OpenAirInterface [Nik+14] or srsLTE [Gom+16] 5G platforms. That means that the interoperability among platforms is not attainable. Furthermore, OpenAirInterface [Nik+14] and srsLTE [Gom+16] can only be operated utilizing expensive wireless hardware [Ett], which makes the extension of testing platforms infeasible. Hence, scaling the network would require a tremendous investment in space and time for research and testing purposes. On the other hand, SD-RAN is still not standardized. Therefore, the extensibility towards new SD-RAN controllers, without changing the system itself, requires an intensive study on the design and implementation of SD-RAN benchmarks.

To overcome this issue, in this chapter we present `MARC`, a novel benchmarking tool for SD-RAN architectures and their controllers. We use `MARC` to measure and analyze two state-of-the-art open-source solutions: `FlexRAN` [Fou+16] and `5G-EmPOWER` [CKR19]. `MARC` extends the controllers' performance analysis of previous works to larger network dimensions (e.g., 5000 network components). By considering the scenario of monitoring applications under fully centralized control, we provide design guidelines to overcome performance bottlenecks for `FlexRAN` with respect to memory resource allocation. Finally, we show that current SD-RAN architectures do not scale, thus we conclude that alternative approaches are necessary to increase the scalability. The detailed contributions of this chapter are summarized as follows:

- We perform a state-of-the-art analysis on SD-RAN based architectures and existing benchmarking tools, while identifying the limitations of existing performance evaluation approaches in terms of scalability in Section 4.2. For this, we analyze the code of the available open-source SD-RAN controllers.
- We design a data plane performance behavior model for SD-RAN controllers to generate UE activation and deactivation events, based on traffic patterns defined by the standardization in Section 4.4.2.4.
- We implement a benchmarking tool based on the design of a novel SD-RAN control plane model and make use of the developed data plane model to test the scalability of open-source SD-RAN controllers in Section 4.4.2.3.
- With the help of `MARC`, we demonstrate the limitations of SD-RAN controllers considering monitoring applications under fully centralized control. Further, we provide an insightful guideline for overcoming performance bottlenecks in Section 4.5. Moreover, we show in Section 4.6 that current SD-RAN architectures with a single SD-RAN controller do not scale and an alternative solution is needed to increase scalability.
- We publish the code of our benchmark [Pap21] to enable researchers to further investigate current and emerging solutions on SD-RAN controllers.

Content and outline of this chapter: Section 4.1 provides a description of existing open-source SD-RAN controllers and demystifies their implementation principles and protocols. Further, related work on controller benchmarking is presented in Section 4.2 for both the core network and RAN. Once general information for SD-RAN controllers is acquired, details are elaborated in Section 4.3 for all the factors that influence SD-RAN control plane load. Given these factors, Section 4.4 introduces `MARC`, our benchmarking tool for SD-RAN controllers and details the measurement setup and output metrics for our analysis. Section 4.5 reports on the main findings of this chapter, whereas the main benefits of `MARC` are highlighted and summarized in Section 4.6. This chapter is based on the following publications:

[Pap+19a] A. Papa, R. Durner, F. Edinger, and W. Kellerer. “SDRBench: A Software-Defined Radio Access Network Controller Benchmark.” In: Proc. of the IEEE Conference on Network Softwarization (NetSoft Workshops). 2019, pp. 36–41. doi: 10.1109/NETSOFT.2019.8806697.

[Pap+21a] A. Papa, R. Durner, E. Goshi, L. Goratti, T. Rasheed, A. Blenk, and W. Kellerer. "MARC: On modeling and analysis of Software-Defined Radio Access Network Controllers." In: IEEE Transactions on Network and Service Management 18.4 (2021), pp. 4602–4615. doi: 10.1109/TNSM.2021.3095673.

4.1 Background on SD-RAN Platforms

In this section, we introduce the concept of SD-RAN. We focus on two open-source SD-RAN controllers in the state of the art which are: `FlexRAN` [Fou+16] and `5G-EmPOWER` [CKR19], while detailing their main functionalities and protocols. For both `FlexRAN` and `5G-EmPOWER`, we provide elaborated details, as it is crucial to the design of our benchmarking tool.

4.1.1 Software-Defined Radio Access Networks

The main concept of SD-RAN lies on the control functionality over the RAN. This implies a full separation of the data and control plane of the RAN, where the control resides in devices known as SD-RAN controllers. In turn, the underlying data plane components (i.e., BSs) communicate with SD-RAN controllers by utilizing control protocols similar to OpenFlow [McK+08], which is one of the most well known SDN protocols in the core network.

A high level overview of the SD-RAN paradigm is given in Fig. 4.1. As presented, SD-RAN controllers abstract the underlying network for the applications. The nature of such applications can vary from monitoring to operational and management. To facilitate the communication between the SD-RAN controllers and the underlying network, BSs are equipped with control components, referred to as SD-RAN agents. For the remainder of this thesis, the terms **agents** and BSs will be interchangeable. Protocol functions within BSs are structured according to the proposed O-RAN splits [O-Ra], namely, CU-CP, CU-UP, DU and RU, as demonstrated in Fig. 4.1. However, for the sake of simplicity, in this thesis we refer to a BS as a single component.

SD-RAN agents understand the SD-RAN protocol and are responsible for translating and enforcing the SD-RAN controller's decisions on the underlying BS devices. While SD-RAN agents reside at the BSs, SD-RAN controllers can be deployed on edge servers and communicate via TCP with the SD-RAN agents. Finally, UEs are served by BSs in order

Table 4.1: Overview of the features supported by `FlexRAN` and `5G-EmPOWER` SD-RAN controllers

Features	FlexRAN	5G-EmPOWER
BS reports	✓	✓
UE reports	✓	✓
Trigger events	✓	✓
Control delegation	✓	✗
Real time controller	✓	✗
Security	✗	✓
Handover	✗	✓
RAN slicing	✓	✓

to enable communication. The block composed of the SD-RAN controller and applications, is referred to in the O-RAN terminology [O-Ra], as near-RT RIC. Furthermore, the near-RT RIC is connected with the orchestration and automation layer in order to provide the design, policy and configuration of the network.

In RAN, the main challenges include an efficient allocation of the scarce radio resources, but also management of handovers and interference. To be able to achieve these goals, efficient scheduling, interference cancellation and handover algorithms are built in the SD-RAN controllers [Fou+16; CKR19; Pap+20b]. A detailed overview of the main features supported by `FlexRAN` and `5G-EmPOWER` is summarized in Table 4.1.

4.1.2 FlexRAN

`FlexRAN` [Fou+16] is an SD-RAN platform built on top of OpenAirInterface [Nik+14], which in turn is an open-source research platform for 4G/5G communication. In this part of the chapter, we initially explain the `FlexRAN` architecture and later the protocol used for the communication between the `FlexRAN` controller and `FlexRAN` BSs (i.e., agents).

4.1.2.1 Architecture

The `FlexRAN` architecture follows the overall SD-RAN concept presented in Fig. 4.1. The summary of `FlexRAN`'s features is portrayed in Table 4.1. `FlexRAN` provides an abstraction layer for applications to be able to monitor, change or adapt several parameters of the network (i.e., power, resource allocation, modulation and coding schemes), based on request, utilizing REST APIs. The platform consists of the master `FlexRAN` controller, and the `FlexRAN` agents. The `FlexRAN` agents, are responsible for enforcing all the control rules with respect to resource allocation, handovers on the underlying physical network. A single dedicated `FlexRAN` controller is envisioned for the control over multiple agents, each one representing

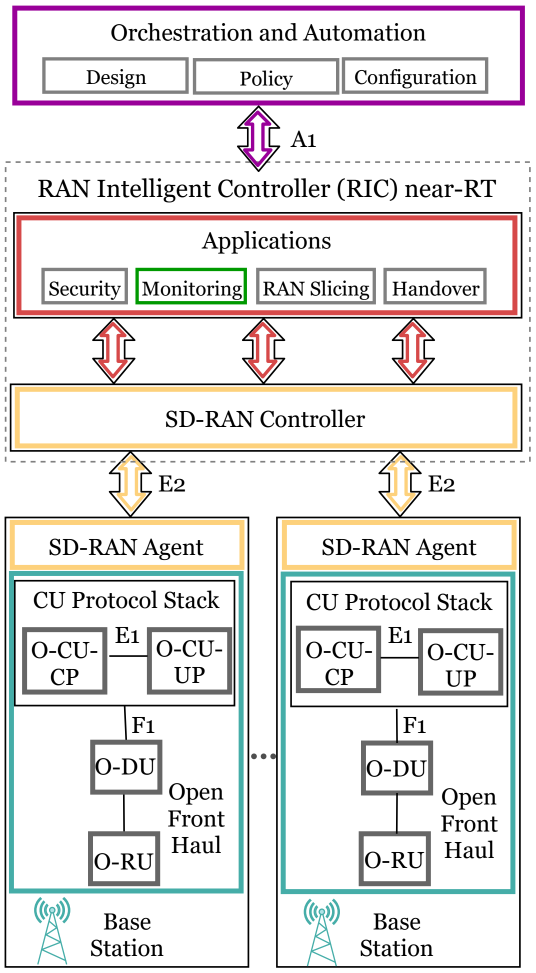


Figure 4.1: High level overview of the SD-RAN paradigm according to the O-RAN terminology. An SD-RAN agent resides at each BS in order to enable the communication with the SD-RAN controller. The SD-RAN controller provides an abstraction overview of the underlying network to the applications and enables them to modify BS's parameters such as power, scheduling, modulation and coding. This block is referred to as near-RT RIC. The near-RT RIC controller is then connected with the orchestration and automation layer, responsible for design and network policy configuration.

a physical BS. However, `FlexRAN` can delegate the control to the local `FlexRAN` agents at runtime to reduce the delay. The communication among them is realized by using the `FlexRAN` protocol.

4.1.2.2 Protocol

The `FlexRAN` protocol defines the set of rules used for the communication in the control plane among the master `FlexRAN` controller and the `FlexRAN` agents. The full flow diagram of the protocol is presented in Fig. 4.2. For `FlexRAN`, the protocol has been identified by details provided in [Fou+16] and by performing analysis on the original code. We have divided the protocol diagram into 5 phases, each of them representing one functionality or event.

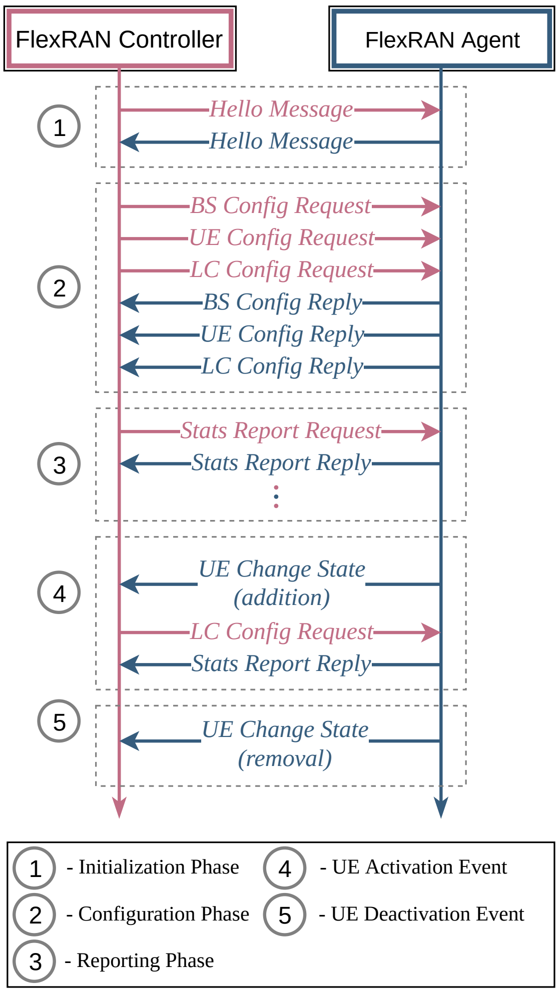


Figure 4.2: Overview of the `FlexRAN` protocol. The protocol is split into 5 phases, each one representing a set of messages exchanged between the `FlexRAN` SD-RAN controller and the `FlexRAN` agent.

Initially, a **hello handshake** message is needed to establish the connection and registration of the agent to the `FlexRAN` controller's database. Further, the `FlexRAN` controller performs a round of requests regarding BS configurations, UE configurations and Logical Channel (LC) configurations. Consequently, the agent replies with all the necessary information back to the `FlexRAN` controller terminating the **configuration phase**. Once the initial setup has terminated, the controller requests a **periodic statistics report** analysis from the agent. The report includes information with respect to the BS and active UEs. Such reports are periodically transmitted with a frequency of 5 ms to provide a real time network monitoring.

Additionally, the `FlexRAN` protocol does not only support periodic events, but it is also triggered by events such as UE activations/deactivations in the underlying data plane. In case of an activation event, the agent notifies the `FlexRAN` controller with a UE change state (i.e., **UE activation**). Immediately, the FlexRAN controller requests information regarding the UE (i.e., LC configuration request). In turn, the agent provides them to the controller and inserts the newly joined UE in the list of active UEs. This list is used to fill in the information for the next periodic report message. Finally, a trigger message can occur also for a UE change state to **UE deactivation**. The FlexRAN controller keeps all the information acquired from its agents to a database known as RAN Information Base (RIB). The RIB is loaded in the memory in order to achieve faster performance. It is updated every $\sim$ 1 ms and it is structured as a forest graph, where each agent is a root of the tree and UEs are the leaves. However, storing such frequent updates in the memory can pose challenges when the network dimensions increase. To support real time operations, FlexRAN is designed such that it decouples the writing and reading of the RIB from the applications. This is achieved by assigning 20% of the processing time to a **RIB updater** and 80% to a **task manager** that handles the applications in a non-blocking mode.

4.1.3 5G-EmPOWER

Similarly to FlexRAN, 5G-EmPOWER [CKR19] is an open-source SD-RAN controller platform existing in the state of the art. It follows the general architecture of the SD-RAN paradigm introduced in Fig. 4.1.

4.1.3.1 Architecture

The 5G-EmPOWER platform provides an abstract view of the network including UEs and BSs to various applications and as such enables the modification of scheduling, power management, handover decisions. The communication between the applications and the 5G-EmPOWER controller is realized by means of REST APIs. According to Table 4.1, for 5G-EmPOWER such applications include monitoring, security, handover, and RAN slicing. An underlying physical infrastructure consists of multiple BSs, where each BS is represented by a 5G-EmPOWER agent. In turn all the information from the 5G-EmPOWER agents are gathered in the 5G-EmPOWER controller.

Differently from FlexRAN, 5G-EmPOWER is not a real time controller i.e., does not adapt to network changes on short time scales in the order of 1 ms. Instead, such adaptations are influenced by instructions sent by the applications. Furthermore, the control of the network cannot be delegated to the local 5G-EmPOWER agents in runtime unlike FlexRAN. These design choices can influence the data plane performance since infrequent updates might not reflect changes in UEs' behavior, which can result in interference or throughput decrease. Nevertheless, in contrast to FlexRAN [Fou+16], 5G-EmPOWER is the only SD-RAN controller

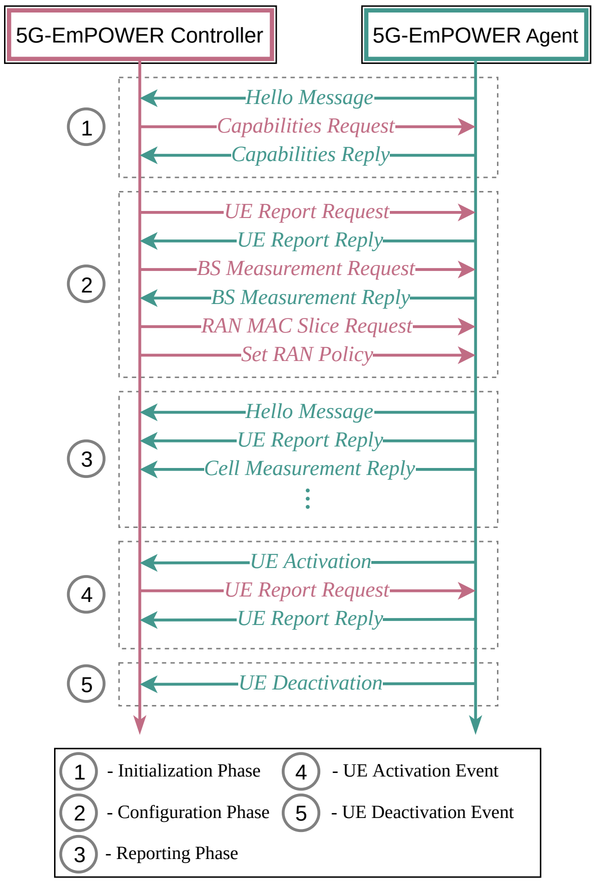


Figure 4.3: Overview of the `OpenEmpower` protocol. The protocol is split into 5 phases each one representing a set of messages exchanged between the `5G-EmPOWER` SD-RAN controller and the `5G-EmPOWER` agent.

that supports security and handover. Finally, the communication in the control plane, among the `5G-EmPOWER` controller and the `5G-EmPOWER` agents is realized by using the **`OpenEmpower`** protocol.

4.1.3.2 Protocol

The communication in the control plane of the `5G-EmPOWER` platform is realized by the **`OpenEmpower`** protocol. Fig. 4.3 provides an overview of the **`OpenEmpower`**. For the `5G-EmPOWER`, the protocol has been identified after tracing the actual implementation in combination with information provided in [CKR19]. We have split the protocol into 5 phases. Similarly to `FlexRAN`, an initialization phase consists of a **hello handshake** between the `5G-EmPOWER` agent and `5G-EmPOWER` controller. Further, the `5G-EmPOWER` controller initiates the **configuration phase** by requesting information with respect to the capabilities of the BS, UE reports as well as initiates BS and UE measurements. Once, the controller gathers all the required information, it enforces the RAN slicing policies. The third phase consists of a **periodic report** phase, which includes hello keep alive messages, as well as UE and BS measurement reports every 500 ms. While such an infrequent update decreases the signaling overhead of the controller, it does not reflect the latest wireless channel characteristics and it can lead to suboptimal performance.

`5G-EmPOWER` supports trigger messages regarding the UE activation and deactivation. Once a UE joins the network, the `5G-EmPOWER` agent sends a trigger **UE activation** message to the controller, which in turn requests a UE report, similarly to the LC configuration message of the `FlexRAN` controller. Immediately, the agent replies back with a UE report reply. Moreover, it adds the newly joined UE to the list of active UEs, which is needed to construct the report messages. Differently from the `FlexRAN` agent, the `5G-EmPOWER` agent does not concatenate all the UEs into a single report message, but instead sends a separate report message for each UE. While this does not have a great impact in a network with low number of UEs (e.g., ~ 10), yet it can pose challenges when the number of UEs increases to realistic deployments (e.g., ~ 5000). Finally, an agent triggers the controller even in the case of a **UE deactivation**. Even for `5G-EmPOWER`, a database is implemented to keep statistics for the newly joined agents and UEs. However, in contrast to `FlexRAN`, in order to increase security, `5G-EmPOWER` only serves those agents whose ID is already registered in the controller's database. For our analysis, a modification on the original database (i.e., addition of agents' IDs) is necessary to enable the communication of the `5G-EmPOWER` controller with our proposed benchmarking tool.

4.2 Related Work

The concept of SDN has recently triggered vast amount of ongoing research in RAN. From an architectural perspective, several conceptual works [Gud+13; Yan+13; ASR15; AWL15; Ber+14b; KL14] address the potentials of SD-RAN and propose techniques to overcome vital RAN challenges such as resource management, coordination, handover and interference cancellation. However, they lack practical implementations.

Recently, a large amount of softwarized prototypes have emerged in the state of the art to fill this void [Fou+16; FMK17; CKR19]. `FlexRAN` [Fou+16] is the first open-source SD-RAN controller platform that provides a separation of the control and data plane in RAN and enables a centralized control functionality. `FlexRAN` is built for OpenAirInterface [Nik+14], which in turn is a software platform that allows for 4G/5G experimentation. `5G-EmPOWER` [CKR19] is another experimental platform built on top of srsLTE [Gom+16] and provides a control and data plane separation for RAN. Finally, `Orion` [FMK17] is an extension of `FlexRAN` and provides an approach for efficient RAN slicing. However, unlike the previous mentioned platforms, Orion is not open source and its potentials cannot be fully exploited by researchers in the field.

The above experimental platforms equip academia with powerful tools and provide the possibility to contribute with practical experiments. This enables faster progress towards satisfying the new 5G requirements [3GPa], and at the same time triggers a closer collaboration between industry and academia. However, while the current evaluation provided for 5G-EmPOWER [CKR19], Orion [FMK17], OpenAirInterface, srsLTE in [Gri+18] and FlexRAN [Fou+16; Pap+19a] is sufficient for academic purposes, the possibility for a commercial use requires further investigation with focus on performance guarantees in realistic network dimensions, which is currently not fully investigated.

In [Gri+18], an evaluation is conducted for OpenAirInterface and srsLTE, highlighting a limited number of 4 UEs on 4G/5G softwarized prototypes [Nik+14; Gom+16; Gar+18a]. This occurs mainly due to design choices or to avoid the purchase of expensive hardware for each UE. Regarding SD-RAN controllers, Foukas et al. [Fou+16] provide an analysis of CPU and memory utilization as well as signaling overhead for FlexRAN. However, only 4 BSs and 16 emulated UEs are considered and thus the real scalability of the controller remains unknown. Recently, a new FlexRAN version has been introduced by Mosaic5G [Mos21], which improves the RAN slicing capabilities and provides handovers. To the best of our knowledge, performance evaluations have not been performed yet compared to the original FlexRAN [Fou+16]. Furthermore, Conorado et al. [CKR19] evaluate 5G-EmPOWER with respect to the CPU utilization with varying number of emulating BSs and UEs for a handover use case. However, implementation details regarding the emulation of the BSs and UEs as well as a signaling overhead analysis are not explicitly provided.

To fill the void, we design MARC, an SD-RAN benchmarking tool able to provide a detailed performance evaluation for FlexRAN [Fou+16] and 5G-EmPOWER [CKR19] considering up to 5000 connected devices. The implementation of MARC is based on similar benchmarking tools for SDN controllers and hypervisors [She; Jar+12; Fou; Ble+19], which have proven to be successful and beneficial for the core network side, yet not available for RAN. MARC distinguishes from existing SDN controller tools, since it includes the protocols of SD-RAN and incorporates specific signaling information unique for RAN. That said, additional mobile protocol stack details have to be implemented.

4.3 SD-RAN Control Plane Modeling Components

In order to provide a fully fledged benchmarking tool for SD-RAN controllers, initially an understanding of all the factors that trigger control plane messages needs to be acquired. The rationale lies in the fact that these are the sources of traffic generation towards the controller. In this section, we present an overview of the main components that influence in the design of the control plane model.

As aforementioned, the SD-RAN controller is the heart of each SD-RAN platform. The controller serves as a connection bridge between the applications and the data plane (i.e., BSs, UEs). Thus, for a lot of applications such as network slicing, MEC [Rig+21; CYR20; Har+20], Cloud RAN [AVK19; AK19; AJK20], a fully capable controller is required. Yet, until now, it is not clear if the available controllers can serve all requests coming from the data plane and the applications in a timely manner. The load on the controller depends on a number of factors as depicted in Fig. 4.4:

1. The number of SD-RAN BSs in the data plane (e.g., `FlexRAN`, `5G-EmPOWER` agents).
2. The amount and type of traffic in the data plane (e.g., UE traffic).
3. The number of the applications and services on the application layer.
4. The type of applications running on the application layer.

Here, we briefly elaborate on the impact of the above mentioned factors on the controller load and behavior. For instance, a high traffic UE load in the data plane can increase the frequency of the communication between the agent and the controller. Similarly, a handover might also trigger reports in order to inform the SD-RAN controller for these events.

From the application layer perspective, the SD-RAN controller should be able to sustain a normal behavior in order to cope with the application requirements. Intuitively, the load of the applications or services that operate on top of such controllers directly affects their performance. Moreover, the nature of applications or services might also play a huge role on the traffic imposed on the controller. For example, if we think of network slicing, constant slicing requests might be triggered by the application layer every time a new slice is initiated or deleted. Moreover, to account for timely monitoring of these slices, a considerable amount of messages between the SD-RAN controller and the application itself might be required.

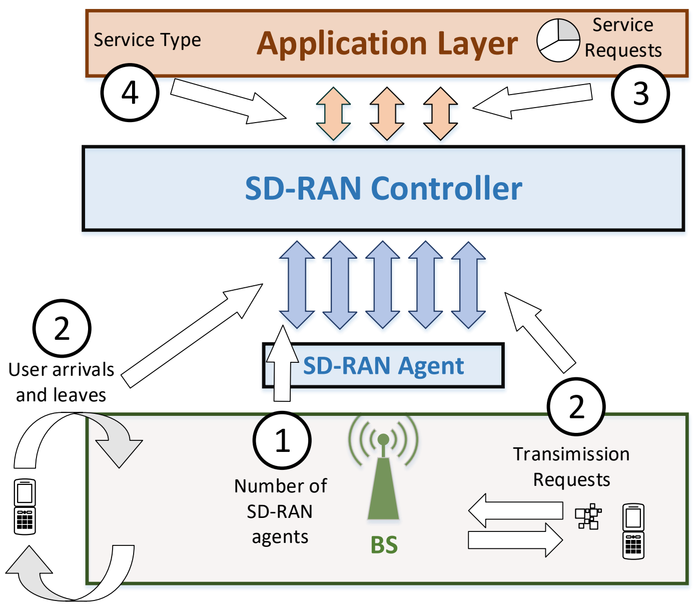


Figure 4.4: Factors that influence SD-RAN control plane modeling. The load on the controller depends on 1) Static BS messages, 2) UE arrivals and departures, 3) Application requests and 4) Application type.

Given the above mentioned factors which can influence the behavior of SD-RAN controllers, a more realistic and complete model should be created. To that end, in the next section of this chapter we demonstrate how the above mentioned factors are included in the design and implementation of `MARC`.

4.4 MARC Design

In this section we present `MARC`, our benchmarking tool for SD-RAN controllers. Initially, we explain the design goals of `MARC` and further we detail the architecture and implementation choices.

4.4.1 Design Goals

The implementation of `MARC` is designed such that it fulfills goals as described below:

- **Ability to provide statistics:** The benchmarking tool must provide statistics with respect to the performance of the SD-RAN controllers considering CPU and memory consumption, throughput as well as latency. This feature enables controller characterization and evaluation of their performance predictability.

- **Scalability:** MARC should be able to provide performance evaluations for SD-RAN controllers considering large number (e.g., > 2000) of connected network components (i.e., BSs, UEs). To that end, the ability to understand whether current controllers can be used in realistic deployments is obtained.

- **Interoperability:** Our benchmarking tool should be compatible with existing open-source SD-RAN controllers. Given the fact that the SD-RAN protocol is not standardized, each SD-RAN platform is tied to its respective data plane system. This means that distinct SD-RAN controllers cannot communicate with other systems. To enable interoperability, a unified tool is needed. MARC offers two SD-RAN protocols as programming libraries that can be used to abstract the data plane for two open-source SD-RAN controllers, which are FlexRAN and 5G-EmPOWER.

- **Extensibility:** Finally, MARC must be extensible for emerging SD-RAN controllers. Even though SD-RAN protocols vary, they still have a specific structure in common (i.e., initialization, statistics requests/reply, trigger messages). In that regard, if the new SD-RAN controller follows either one of the already available protocol libraries supported by MARC, no changes are needed for the operation. Alternatively, an adaptation of the new controller's protocol to fit the common protocol structure is required. This can be easily achieved by extending the current programming libraries without re-programming the system itself.

4.4.2 Architecture and Implementation

As described in Section 4.1, FlexRAN and 5G-EmPOWER have different protocols and design characteristics. For example, the statistics report frequency and concatenation of BS and UE report messages. Thus, in our benchmarking implementation we need to take this information into account and create separate functions for each SD-RAN agent. MARC contains more $\sim$ 3000 lines of new code. All the functionalities within an agent are implemented in a non-blocking manner using *asyncio* [Pyt19a], which is a Python-based package that enables a concurrent execution of the processes.

In this part, we initially present the architecture overview of MARC. Furthermore, we describe the message libraries required for our implementation and detail the control plane model functionalities of our benchmarking design. In order to emulate a realistic environment, apart from the SD-RAN agents representing the BSs, UEs should be included in our model. However, most of the existing prototype platforms built on top of OpenAirInterface [Nik+14] and srsLTE [Gom+16] are limited to a maximum of 4 UEs [Nik+14; Gom+16; Gri+18;

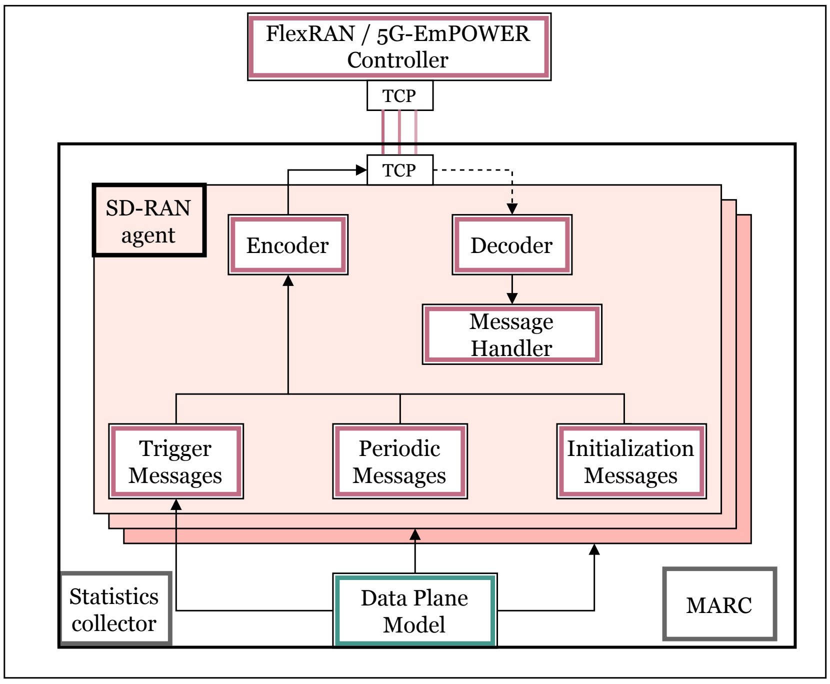


Figure 4.5: Architecture overview of the benchmarking tool. It portrays the model and implementation of the SD-RAN agents, each one for distinct SD-RAN controllers, `FlexRAN` and `5G-EmPOWER`. The control plane functionalities are depicted with red color, whereas green color depicts the data plane model function. The communication with the SD-RAN controllers is realized by means of standard TCP using *asyncio*, which provides an asynchronous operation.

Gar+18a]. Hence, due to such design limitations and to avoid the purchase of expensive hardware [Ett], we provide a performance model for SD-RAN data plane traffic based on the literature [Tri+18; AP19; YKS14; Bro16].

4.4.2.1 Architecture

The overview of `MARC`'s architecture is depicted in Fig. 4.5. Both `FlexRAN` and `5G-EmPOWER` controllers are connected to `MARC` by means of standard TCP. We distinguish between control plane functionalities depicted with red color, representing the SD-RAN agents, whereas green color portrays the data plane model function. A statistics collector is implemented to enable the gathering and analysis of the results obtained from the measurements. In order to test the scalability of `FlexRAN` and `5G-EmPOWER`, `MARC` can generate multiple threads of SD-RAN agents, each with a distinct data plane traffic and TCP port. Moreover, `MARC` supports two operation ways. In the first one, all SD-RAN agents can run centralized on a single server. Alternatively, SD-RAN agents are distributed on different machines. However, in this chapter, we demonstrate results for the former approach, where all SD-RAN agents are centralized on a single server.

4.4.2.2 Message Libraries

In order to provide compatibility and interoperability among FlexRAN and 5G-EmPOWER protocols, MARC provides the SD-RAN platform message specific functionalities as programming libraries that can be used to encode/decode and handle messages as follows:

- **Encoding/Decoding**: MARC implements *encoding* and *decoding* for both **FlexRAN** and **OpenEmpower** protocols as suggested by the original platforms. On the one hand, for the FlexRAN protocol we utilize **Google Protocol Buffers** [Goo19], which makes use of the mechanism of Protobuf to serialize the structured message types for the communication. On the other hand, for 5G-EmPOWER we utilize the Construct python library [Pyt19b].
- **Message handler**: The message handler is the core functionality of our benchmarking tool, as it handles all the replies from/towards the controller and behaves according to the protocols. For each SD-RAN agent, the message handler is adapted to act similarly to **FlexRAN** and **OpenEmpower** protocols. Again due to lack of a standardized protocol, the message handler is offered as a library that can be interchanged among the two protocols.

4.4.2.3 MARC Control Plane Modeling

The control plane modeling for MARC includes the implementation of functionalities for the agents of FlexRAN and 5G-EmPOWER platforms. In turn, the FlexRAN and 5G-EmPOWER controllers, which are the heart of the respective architectures are benchmarked by our tool. Current SD-RAN controllers are tied to OpenAirInterface and srsLTE 4G/5G platforms, accordingly. Moreover, the existing operation mode is only available while utilizing expensive wireless hardware, which makes the extension of testing platforms infeasible. In order to overcome this issue, we emulate the FlexRAN and 5G-EmPOWER agents with focus on the control plane functionalities. That entails BS and UE signaling reports, BS initialization messages, BS configuration messages and UE activation and deactivation messages as elaborated in 4.1.2.2 and 4.1.3.2, respectively.

We enhance the operation towards a multi-threading mode, where each agent is accompanied with a single thread to test the controllers' scalability with minimal computing resources. Our main goal is to benchmark SD-RAN controllers considering a high network load. Thus, currently, MARC deploys the extreme case that the network control is delegated fully to the SD-RAN controllers and is not carried out from the BSs. In terms of scalability, this demonstrates a corner case, which can be dynamically changed by offloading control between the SD-RAN controllers and BSs. However, MARC can be extended to enable such a configuration.

Initially, that will require the creation of control delegation events on our data plane model depending on 3GPP distributions or realistic traces, following a similar approach to the activation/deactivation events presented later in the chapter. Furthermore, modifications are also needed in the message handler to cater for such events in our control plane model. Nonetheless, the effort for including this processing is minimal as it is equivalent to removing BSs and their respective UEs from our measurements dynamically. The main control plane functionalities can be summarized as follows:

- **Trigger message generator:** These type of messages are categorized as asynchronous messages and are supported both by `FlexRAN` and `5G-EmPOWER`. They generally consist of new UEs joining or UEs leaving the network. The information obtained from the data plane model given the UE activation/deactivation is reported to the message handler.
- **Periodic message generator**: Synchronous message type, which is in charge of generating periodic reports either for the UE or the BS behavior. For `FlexRAN` there is more frequent periodicity that allows operation for real time applications (∼ 50 ms). In contrast, for `5G-EmPOWER` such reports are less frequent (∼ 500ms) since the real time adaptation is not the focus.
- **Initialization message generator**: Includes all synchronous messages that are exchanged between the SD-RAN controller and its respective agent at the beginning of the connection. As shown earlier in Fig. 4.2 and Fig. 4.3, such messages consist of hello messages, LC configuration, UE and BS configuration.

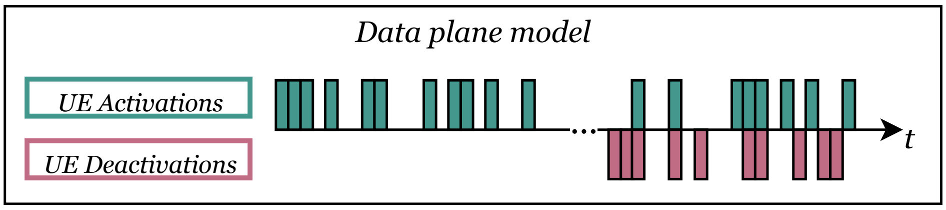


Figure 4.6: Data plane model used for the UE traffic generation. It emulates the number of UEs entering or departing the network, where each event is presented as a tuple consisting of an event *type* and a *timestamp*. The creation of the events is based on an SD-RAN data model proposed in this chapter, where the parameters were taken from literature.

4.4.2.4 Data Plane Modeling

The data plane model is used to emulate UE behavior for each SD-RAN agent (i.e., BS). We base our model on 5G specifications [3GP19b], where services are separated into eMBB, mMTC and URLLC. The aforementioned services have heterogeneous characteristics including cell area, UE density, service type, inter-activation time and duration. Our model is built for any service type, but in our scenario we show results for eMBB, which consists of UEs requiring high throughput.

Once the thread of an SD-RAN agent is initialized, according to our data plane model a series of UE events are generated. These events represent UE activations and deactivations as portrayed in Fig. 4.6, where green color denotes the UE activation and red color the UE deactivation. Each event is accompanied with a *timestamp*. In our measurements, the UE activations follow an exponential distribution with a mean of 10 s per activation, whereas UE deactivations follow a deterministic distribution with a value of 500 s. Details on the UE characteristics and requirements are taken from [Tri+18; AP19; Bro16]. `MARC` allows for a reconfiguration of those values according to the scenario. Since extensibility of the benchmarking tool is one of our main design goals, `MARC` can be connected to any type of UE simulator that generates UE activation and deactivation events in real time in order to test SD-RAN controllers. The only requirement is that events must be accompanied with a type and timestamp.

For each generated UE, resembling a realistic scenario, characteristics with respect to the channel quality, number of requested resources, number of allocated resources, details with respect to the transmission power, modulation and coding scheme are provided. This information is then transmitted to the respective controller as part of the periodic message generator. Considering monitoring applications, the information within the periodic messages with respect to the control plane load is not relevant, as the structure of the messages is fixed by the SD-RAN protocol and it only depends on the number of UEs in the network. However, for applications such as scheduling or RAN slicing this information is crucial, as it influences the resource allocation. In that regard, generating realistic UE statistics becomes mandatory. Although in this chapter we demonstrate results for a monitoring application, `MARC` allows for the deployment of various configurations. In the periodic message construction, each UE's data can be populated by generating statistics from 3GPP distributions or real channel traces and encoded accordingly. This process generates unique information for each UE and can be utilized as input for the SD-RAN controllers. Thus, alleviating the extensibility towards more complex, but vital RAN applications such as scheduling and RAN slicing.

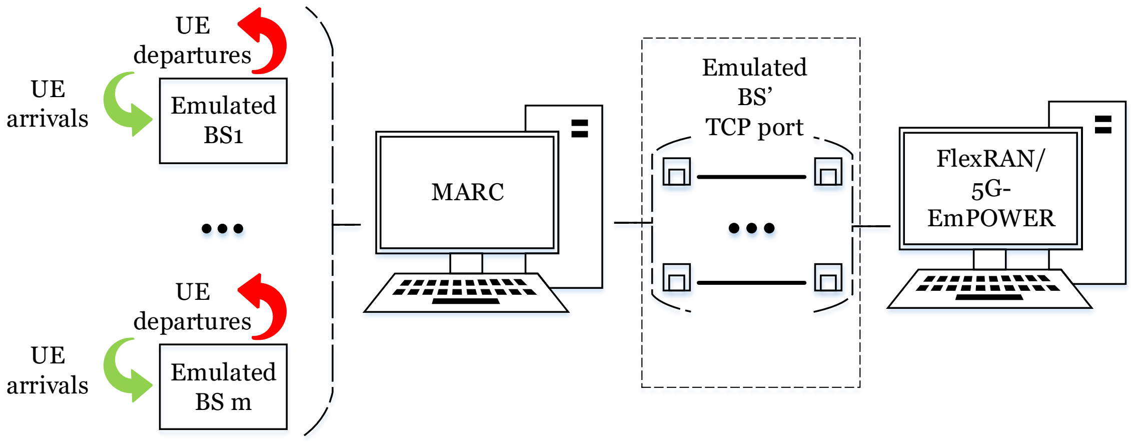


Figure 4.7: Measurement setup diagram. It portrays the setup for the `FlexRAN` and `5G-EmPOWER` controllers running on separate Personal Computer (PC)s. Moreover, `MARC` runs in another PC. `MARC` emulates the BSs as well as the UEs for the evaluation. Each UE is generated according to data acquired from literature and is accompanied with information with respect to channel statistics, transmission power, modulation and coding scheme information, resembling a realistic scenario.

4.5 Performance Evaluation

In this section, we describe the measurement setup and reveal our main findings. Furthermore, we provide useful guidelines to overcome performance bottlenecks for SD-RAN controllers considering monitoring applications under a fully centralized control plane. To that end, we identify implications with `FlexRAN` when increasing the network dimensions and propose `AltFlexRAN` to fix them. In principle, for `AltFlexRAN`, the allocation of the resources among the **RIB updater** and **task manager** of `FlexRAN` is adapted according to the application to avoid read/write memory issues. In the remainder of this section, we will provide in detail guidelines for the adaptations of `AltFlexRAN`.

We evaluate the SD-RAN controllers using performance indicators similar to existing SDN controller and hypervisor benchmarks [She; Fou; Jar+12; Ble+19]. Initially, we measure the controller delay with respect to **SD-RAN initialization**. Moreover, we evaluate the **CPU** and **memory utilization**, where the python library *psutil* [Pyt19c] is exploited for the analysis. To test the controlling overhead of the SD-RAN controllers, we measure the control message throughput both towards and from the controller (i.e., **transmission & reception rate**) to identify the stable operational region. We show the results both in msg/s and kbps to better highlight the signaling overhead since `5G-EmPOWER` sends more, but smaller messages compared to `FlexRAN`. A monitoring application runs on the SD-RAN controllers. Each measurement is repeated 10 times, each consisting of 10 minutes; the values are recorded every second. In order to exclude the transient phase of the measurements, we remove the 50 initial and last measurement points from our analysis, i.e., 50 seconds. In our measurements, the data plane traffic for the UE activation time is modeled with an exponential distribution with a mean value of 10 s per activation. In turn, the UE deactivation time is modeled with a deterministic value of 500 s.

Here, we want to stress that our results are similar to the ones reported from the original `FlexRAN` [Fou+16] and `5G-EmPOWER` [CKR19] works regarding a small amount of UEs and agents, thus demonstrating the effectiveness and credibility of our benchmark. Moreover, we recall that the terms agent and BS is used interchangeably in this thesis and they have the same meaning. In line with these results, in the remainder of this section we present the extension towards larger network dimensions as well as additional results for the signaling overhead and SD-RAN agent initialization delay.

4.5.1 Measurement Setup

The experimental setup for the measurements consists of 2 Desktop PCs, each one equipped with an Intel(R) Core(TM) i5-3470 CPU, containing 4 physical CPU cores at frequency 3.2 GHz and 8 GB RAM. Both PCs run Ubuntu 18.04.2 LTS with 4.15.0-58-generic kernel. One of the PCs is running the SD-RAN controller of choice, either `FlexRAN` or `5G-EmPOWER` based on the open-source code found on the original git repositories [Mos21] for `FlexRAN` and [Rob19] for `5G-EmPOWER`. Whereas the other PC operates the modeled SD-RAN agent.

The detailed measurement diagram is portrayed in Fig. 4.7. `MARC` emulates the BSs and UEs in the network depending on the test parameters of choice. For each emulated BS, a distinct thread and TCP port is created to distinguish among packets sent from alternative BSs. Within a BS thread, UEs are generated according to data from 3GPP standardization [3GP19b]. Each UE carries important characteristics such as channel quality, transmission power and modulation and coding information. Moreover, `MARC` utilizes the python library of *asyncio* [Pyt19a] to guarantee asynchronous operation among the threads.

4.5.2 SD-RAN Agent Initialization Delay

Our initial evaluation consists of controller delay measurements regarding the initialization of SD-RAN BSs (agents). These results are particularly interesting for 5G/6G networks with stringent delay requirements, as they illustrate the ability to maintain next generation applications.

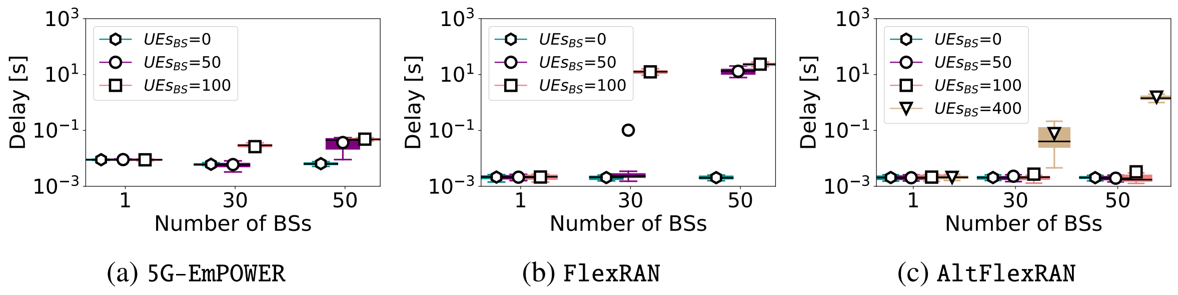


Figure 4.8: Performance evaluation: Initialization delay in seconds. 5G-EmPOWER demonstrates an SD-RAN agent initialization delay of ∼ 10ms, whereas the delay for FlexRAN increases up to 10s. AltFlexRAN, however can sustain up to 50 BSs with 100 UEs each, experiencing up to 1s delay.

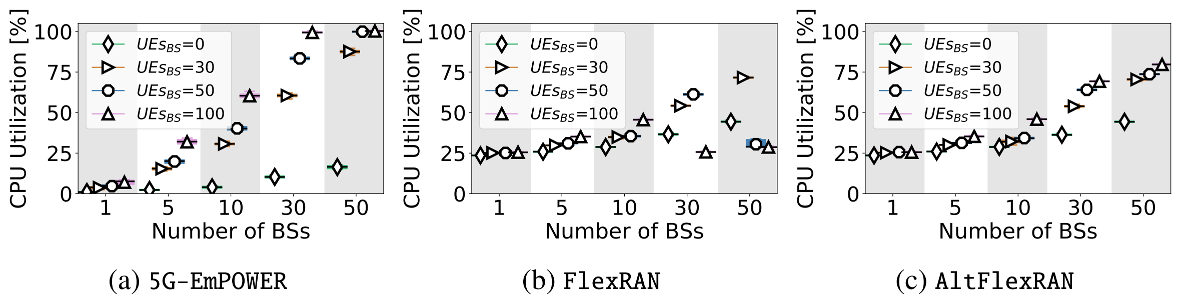


Figure 4.9: Performance evaluation: CPU utilization [%]. The CPU utilization is increasing almost linearly for all SD-RAN controllers. However, for FlexRAN, the CPU utilization decreases when more than 30 SD-RAN agents are served. AltFlexRAN overcomes this bottleneck by assigning effectively resources between the RIB updater and task manager.

We first initiate *m* BSs, each one serving *t* UEs. Once all BSs have finalized their initialization phase, they only send periodic statistic reports. In order to measure the initialization delay of the newly joined SD-RAN BS in the system, we start the *m+1* BS and record its initialization delay. We conduct 50 experiments while varying the previously initiated SD-RAN BSs and UEs generated for each SD-RAN BS. The results are illustrated in Fig. 4.8. The x-axis represents the number of *m* initiated BSs running *t* UEs each, whereas the y-axis the recorded delay in seconds. As denoted by Fig. 4.8a, for the 5G-EmPOWER, the initialization delay remains within reasonable values ∼ 10 ms up to the case when the CPU utilization saturates (i.e., 30 BSs, 100 UEs per BS). In such a case, we notice a drastic increase of the delay up to 50 ms. For the original FlexRAN controller the results are presented in Fig. 4.8b. For FlexRAN we observe a similar experienced delay up to 30 BSs ≤ 10 ms. When the number of BSs increases further, up to 50, the delay increases drastically ≥ 10 s. We will identify the sources of those performance changes in detail in the remainder of this section. We repeat the experiment for the proposed AltFlexRAN and report the results in Fig. 4.8c. Based on MARC's guidelines we are able to contain the SD-RAN initialization delay to 1s even for 50 BSs and 400 UEs per BS.

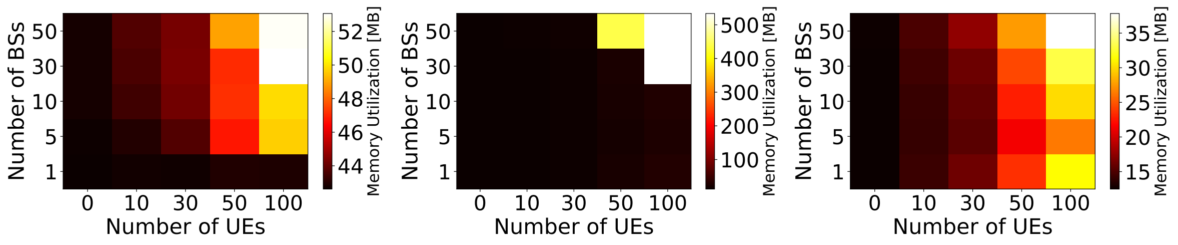
4.5.3 CPU Utilization

The main findings for the CPU utilization are shown in Fig. 4.9. The x-axis represents the number of SD-RAN BSs, while for each SD-RAN BS various UE configurations are denoted. The y-axis depicts the CPU utilization in percent % and it is averaged for all the 4 physical cores of the setup.

As we can observe from Fig. 4.9a, the CPU utilization of the `5G-EmPOWER` controller is increasing almost linearly with respect to the number of BSs and UEs. Similarly to the results reported in [CKR19] with hardware for 1 SD-RAN agent (i.e., BS) and 2 UEs, our tool records CPU consumption of ~ 2%. However, the CPU utilization saturates for the configuration of 30 BSs and 100 UEs per BS and reaches 100%. We observe an identical behavior also for the configuration of 50 BSs and for more than 50 UEs per BS. When no UEs exist in the system (i.e., only BS reports), the CPU load does not increase more than 20% up to 50 considered BSs. But, once UEs are served, the CPU load increases drastically.

Similarly to `5G-EmPOWER`, the CPU load for varying number of BSs and UEs is demonstrated for `FlexRAN` in Fig. 4.9b. Even in this case, `MARC` records comparable results with `FlexRAN` using hardware for 1 SD-RAN agent and 2 UEs of ~ 20%. Even for `FlexRAN`, we notice an increase of the CPU with increasing number of BSs and UEs. Differently from the `5G-EmPOWER`, we notice a CPU drop when reaching a high load (i.e., ~ 30 BSs and 100 UEs per BS). Such a drop is also observed for the case of 50 BSs when the number of UEs increases further than 50 per BS. Hence, we conclude that `FlexRAN` cannot support effectively more than 30 BSs. For the `5G-EmPOWER` controller, the fast CPU increase compared to `FlexRAN` is clear since for each newly added UE in the system, a distinct UE report has to be sent periodically to the controller. This periodicity corresponds to 500ms. For the `FlexRAN` controller, on the other hand, the UE reports are co-allocated with the BS reports and as such the increasing number of UEs does not affect the number of reports as much. This is notably visible in Fig. 4.9b, where even when no UEs are present in the system the CPU load is considerably high. It reaches up to 43% when serving 50 BSs and no UEs, while a utilization of ~ 22% is observed for the initial scenario with a single BS and no UEs.

Here, we again stress that unlike `5G-EmPOWER`, `FlexRAN` is a real time controller. That entails that `FlexRAN` is designed such that it reflects the latest statistics from the data plane (i.e., wireless channel) in order to adapt the resource allocation (i.e., RAN slicing). Therefore the UE and BS reports are transmitted from the BS to the SD-RAN controller more frequently. This occurs every 50 ms in our case. Nonetheless, even though the reports for `FlexRAN` are being sent more frequently, we observe that for the same scenarios, the CPU utilization of



(a) `5G-EmPOWER` (b) `FlexRAN` (c) `AltFlexRAN`

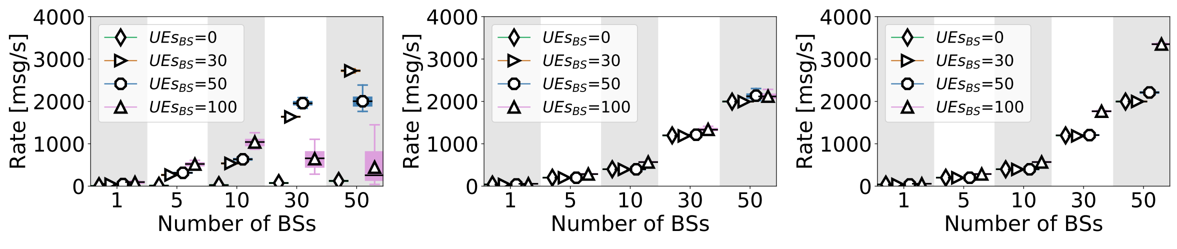
Figure 4.10: Performance evaluation: Memory utilization [MB]. `5G-EmPOWER` demonstrates a normal memory behavior, while `FlexRAN` increases the memory consumption with increasing load. We fix this issue in `AltFlexRAN` by assigning more resources to the RIB updater.

`FlexRAN` unlike the `5G-EmPOWER` does not reach 100%. The rationale behind this behavior is related to the **RIB updater** function of `FlexRAN` that was earlier detailed in Section 4.1.2.1. The **RIB updater** is responsible for writing and reading `FlexRAN` agent-related information to the controller's database. The original `FlexRAN` solution suggests an allocation of 20% of the processing time to the **RIB updater**, whereas 80% of the processing time is associated to a **task manager** function in charge of handling the applications.

Based on the insights provided by `MARC`, we adjust the distribution of the resources among the **RIB updater** and **task manager** to adapt to the monitoring application demands i.e., high frequency of statistic reports. Hence, we repeat the experiments for `AltFlexRAN` by assigning 80% of the time to the **RIB updater** to write/read to/from the RIB and 20% of the time to the **task manager**. As it can be observed in Fig. 4.9c, the CPU utilization for the `AltFlexRAN` controller does not drop anymore in a highly loaded scenario, instead it increases. Therefore, `AltFlexRAN` demonstrates a more stable behavior than `5G-EmPOWER`, even though the periodicity of statistics reports is $\sim 50\,\text{ms}$. Here we can conclude that concatenating the UE and BS reports using the serializing techniques of Google Protocol Buffers [Goo19] consumes less CPU and it is a better design choice for SD-RAN controllers.

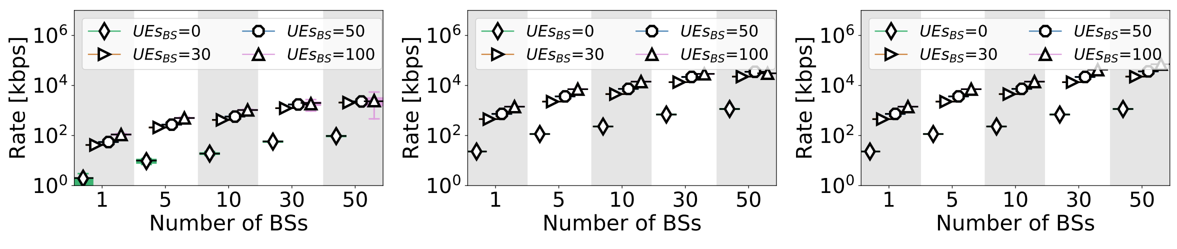
4.5.4 Memory Utilization

In line with the CPU utilization, we further perform evaluations regarding the memory footprint. Ideally, a well-developed program should not demonstrate drastic memory increase even in a highly loaded scenario. The main results for the SD-RAN controllers regarding the memory utilization are shown in Fig. 4.10, where the x-axis represents the number of UEs per BS, whereas the y-axis the number of SD-RAN BSs. The results are represented in MB. As denoted by Fig. 4.10a, `5G-EmPOWER` experiences a normal behavior of the memory utilization. Although the memory utilization increases with higher number of BSs and UEs, it does not



(a) `5G-EmPOWER` (b) `FlexRAN` (c) `AltFlexRAN`

Figure 4.11: Performance evaluation: Reception rate of SD-RAN controllers [msg/s].



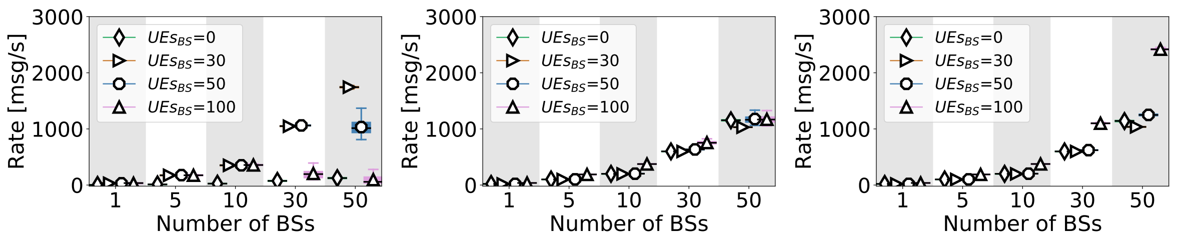
(a) `5G-EmPOWER` (b) `FlexRAN` (c) `AltFlexRAN`

Figure 4.12: Performance evaluation: Reception rate of SD-RAN controllers [kbps].

exceed 110 MB. Alternatively, `FlexRAN` demonstrates an unexpected behavior. As portrayed in Fig. 4.10b, for a high number of BSs and UEs, the memory utilization keeps increasing by reaching an average of 1 GB for the measurement time (i.e., 10 minutes). However, we note that if not interrupted, the memory can increase up to 100% and therefore even crash the controller.

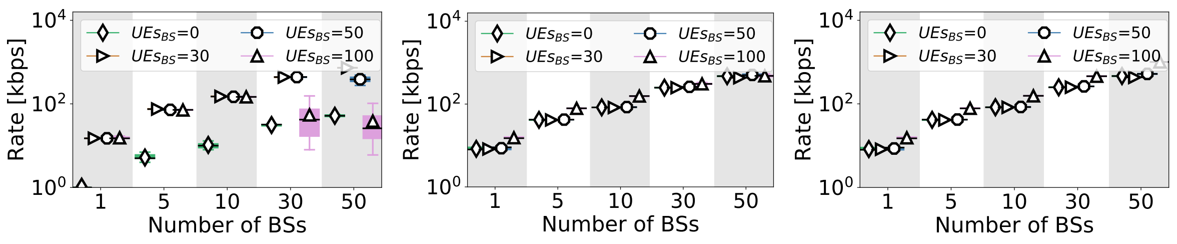
While at first sight this may indicate a memory leakage, after an analysis of the original `FlexRAN` code, we identify that the reason behind this behavior is the processing time allocation of the **RIB updater**. In line with the CPU measurements, when a high load is reached, 20% of the processing time initially assigned to the **RIB updater** is not sufficient to sustain a normal behavior. If we recall the original `FlexRAN` logic elaborated in Section 4.1.2.1, the RIB is loaded directly in the memory for faster performance. The increase of the memory indicates that the **RIB updater**, while it can write all the new information retrieved from the BSs, it does not have sufficient time to process them. Thus, the memory keeps increasing. To overcome this issue, in `AltFlexRAN`, we switch the processing time of the **task manager** and **RIB updater**. The memory results for the `AltFlexRAN` are presented in Fig. 4.10c. As we can observe, the `AltFlexRAN` solution is operating in a normal manner and the memory footprint remains within reasonable values ~ 75 MB.

As a conclusion, we can say that for `FlexRAN` a reasonable allocation of the RIB resources depending on the application can overcome potential performance bottlenecks.



(a) `5G-EmPOWER` (b) `FlexRAN` (c) `AltFlexRAN`

Figure 4.13: Performance evaluation: Transmission rate of SD-RAN controllers [msg/s].



(a) `5G-EmPOWER` (b) `FlexRAN` (c) `AltFlexRAN`

Figure 4.14: Performance evaluation: Transmission rate of SD-RAN controllers [kbps].

4.5.5 Transmission and Reception Rate

In this subsection, we conduct experiments regarding the transmission and reception rate of the SD-RAN controllers and we quantify their signaling overhead. In that regard, we measure the amount of packets and bytes transmitted and received by the SD-RAN controllers while monitoring the IP tables of the PC where the controller is running. To achieve this, we create two IP table rule chains, one for the reception and one the transmission and capture only the packets destined for our application.

Fig. 4.11a and Fig. 4.12a illustrate the number of packets and bytes that the `5G-EmPOWER` controller processes with respect to the number of BSs and UEs. Although both rates are linearly increasing with increasing number of UEs and BSs, at some point (i.e., 30 BSs) we notice a decreasing trend. This happens for the cases when the controller is highly utilized, as a result of packet drops and potential TCP effects. The same experiments are repeated for the `FlexRAN` controller and `AltFlexRAN`. As we can see from Fig. 4.11b, Fig. 4.12b, Fig. 4.11c and Fig. 4.12c the controllers' reception rate increases with the number of BSs and UEs. However, for `AltFlexRAN` the amount of packets and bytes is higher than `FlexRAN`: 2500 compared to 1100 msg/s, since we extend the effective operational region. Intuitively, a similar trend follows for the reception in bytes. Due to the aforementioned design choice differences between `FlexRAN` and `5G-EmPOWER`, the number of packets received by the `5G-EmPOWER` in a stable operation region (i.e., up to 30 BSs, 50 UEs per BS) is higher compared to `FlexRAN`:

2100 compared to 1400 msg/s. Yet, the amount of bytes recorded for `FlexRAN` is higher compared to `5G-EmPOWER`: 70 Mbps compared to 1 Mbps. This is explained due to the fact that `FlexRAN` agents concatenate messages together and the size of the packets is higher compared to the `5G-EmPOWER`'s BSs. The signaling overhead recorded for `FlexRAN`, although higher compared to `5G-EmPOWER`, still consumes less CPU resources than `5G-EmPOWER`. We again conclude that Google Protocol Buffers [Goo19] are more suitable for SD-RAN.

Compared to controllers' reception rate, the transmission rate is not so demanding since the controllers only reply back to the report statistics with an ACK. For instance, for `FlexRAN` the decrease is almost 2 orders of magnitude: 70 Mbps reception compared 1 Mbps transmission rate. A similar trend is observed also for the all the tested controllers. The results of our analysis are portrayed in Fig. 4.13 and Fig. 4.14.

Table 4.2: Observations for the studied SD-RAN controllers. A single SD-RAN controller approach can not guarantee low latencies for more than 2000 devices.

Controllers	**Scalability**	**Signaling Overhead**	**Initialization Delay**
`5G-EmPOWER`	~ 30 BSs, 50 UEs	~ 1 Mbps	~50 ms
`FlexRAN`	~ 30 BSs, 50 UEs	~ 20 Mbps	~10 s
`AltFlexRAN`	~ 50 BSs, 100 UEs	~ 70 Mbps	~1 s

4.6 Summary

While we demonstrated the performance analysis of `FlexRAN` and `5G-EmPOWER`, we discuss whether our benchmarking tool generalizes to other SD-RAN controllers. In principle, `MARC` is extensible and can be used for any SD-RAN controller that follows the general SD-RAN paradigm explained in Section 4.1 and utilizes one of the considered SD-RAN protocols with no additional changes required. Otherwise, changes are needed in the protocol of the added controller, such that the latter fits to the common SD-RAN protocol structure. Nonetheless, this can be achieved by extending the current programming libraries of `MARC` without altering the benchmark's system itself.

Furthermore, we discuss the validity of our benchmarking tool considering the UE and SD-RAN agent emulation. Again we stress that the results measured by `MARC` with respect to CPU, memory utilization and signaling overhead are in line with the results presented for `FlexRAN` [Fou+16] and `5G-EmPOWER` [CKR19] for a small amount of BSs and UEs utilizing hardware. Thus, demonstrating credibility for our measurements.

In this chapter, we evaluated `FlexRAN` and `5G-EmPOWER` for a monitoring application. However, given the capabilities of the aforementioned controllers and the necessity for more complex operations in 5G/6G networks, evaluations with respect to applications such as scheduling and RAN slicing become crucial. Here, we discuss whether `MARC` is suitable for providing extensibility for such scenarios and applications. Generally speaking, a real implementation of an SD-RAN controller for performing tasks such as scheduling or RAN slicing requires information with respect to the UE characteristics. `MARC` constitutes the basis to enable such complex tasks by introducing the periodic report messages as an input to the controllers. Thus, if applicable by the SD-RAN controllers, our benchmarking tool can provide measurements similarly to the monitoring use case. Note that due to the high complexity of such tasks, the results provided for the monitoring application may vary and a fewer number of UEs and BSs might be supported by those controllers. Similarly, the improvements introduced by `AltFlexRAN` for the RIB updater and **task manager** will vary. In that regard, a more dynamic adaptation between the resources allocated for either RIB updater or **task manager** is mandatory to achieve better performance. Alternatively, variations in the presented results for `FlexRAN` could be observed when testing the recent version introduced by Mosaic5G [Mos21]. Given the improved API [Mos20], the number and rate of statistics delegated to the SD-RAN controller can be adapted at runtime according to the wireless channel variations. Utilizing sophisticated algorithms this could lead to serving more BSs and UEs similar to `AltFlexRAN`.

Finally, we highlight the stable operational region of the considered SD-RAN controllers in Table 4.2. Notably, the single SD-RAN controller approach cannot sustain the network load in an urban dense scenario (i.e., > 5000 devices). In that regard, a distributed control plane becomes a must. Whereas this approach may balance the load of the network, it still poses significant design challenges as it requires extra signaling overhead for the inter-controller communication. Therefore, in the next chapter of this thesis, we focus explicitly on providing answers with respect to QoS performance on a distributed SD-RAN environment. Moreover, due to the fact that the UEs generated for the data plane modeling in this chapter are emulated, the real performance on the UE side could not be captured. Therefore, Chapter 5 provides a mathematical model and measurement campaign to evaluate the performance in the data plane for real users, given irregularities of SD-RAN controllers in the control plane in more detail based.

Chapter 5

Quality of Service Provisioning in Software-Defined Radio Access Networks

In the previous chapters of this thesis, we highlighted the fact that compared with previous generations of cellular networks, 5G manifests the flexibility to support a wide range of heterogeneous applications, control and data plane separation as well as the introduction of a service-oriented architecture [3GPa]. To cater for the aforementioned requirements, SDN has become unavoidable in the core network [Jin+13; CYY16; Afo+18]. Similarly, for the RAN counterpart, softwarization [Fou+16; Bon+21a] and virtualization [FR21; Gar+21; Val20] are envisioned as the main drivers to fulfill the anticipated 5G design principles.

As mentioned in Chapter 4, `FlexRAN` [Fou+16] and `5G-EmPOWER` [CKR19] are the first open-source SD-RAN prototypes that provide a separation of the control and data plane of traditional RANs, where the control is shifted from BSs towards centralized entities referred to as **SD-RAN controllers**. In such a centralized setup, SD-RAN controllers maintain a broader network view while updating their knowledge by receiving **control messages** which contain BS and UE statistics. Based on these messages, **control decisions**, concerning MAC scheduling, are enforced at the BSs. Exploitation of the wide network knowledge leads to improved performance as it allows for optimal management decisions.

Yet, as shown in the previous chapter, SD-RAN controllers become a single point of failure and a potential bottleneck. This phenomena is certainly an issue when the number of RAN elements, i.e., BSs and UEs increases drastically. This occurs as the number of control messages traversing the network towards the controller and back to BSs in that case explodes. Such an occurrence can be observed in Fig. 5.1 for measurements performed with the `FlexRAN` controller. The x-axis represents the number of connected BSs, for various numbers of UEs served by BSs, and the y-axis depicts the control overhead in Mbps. Notably, the control

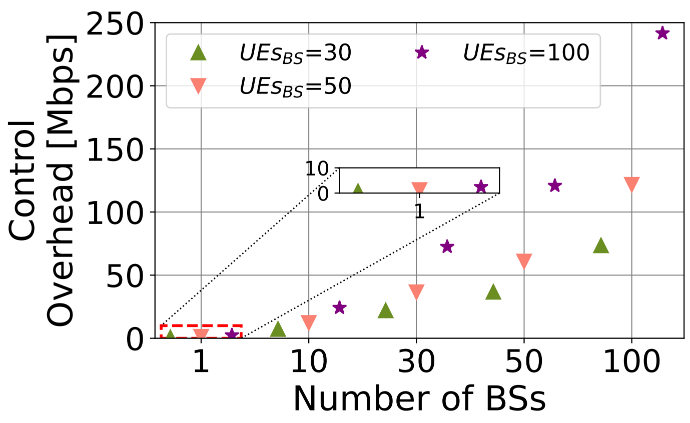


Figure 5.1: SD-RAN controller overhead generated by control messages with respect to UEs and BSs in Mbps.

overhead increases faster than linearly with the number of network elements. E.g., when the number of RAN elements reaches 10000, the control overhead takes up to 250 Mbps, which is a considerable burden.

When considering 5G networks with thousands of connected devices [LJ16; Oug+19], there are two main problems that arise. The first lies in the inability of the existing solutions to provide any performance prediction for networks of large dimensions. The second is concerned with the development of architectures and methods that guarantee the predicted level of service.

In order to be able to tackle these issues, initially we need to gain a deeper understanding of the existing SD-RAN solutions in the literature and provide insights on performance predictability, which is a prerequisite to construct new models and architectures. While the former was achieved by introducing `MARC` (Section 4.4), an important part of the network, UEs, have not been considered in depth so far. The reasonable question which then arises is *how do SD-RAN overloaded controllers affect the UEs in the data plane, and what happens with the overall throughput*? To the best of our knowledge, this question is left unanswered in the literature so far.

In order to fill this void in the state of the art, in this chapter, initially, we investigate methods that predict the throughput in the data plane depending on imperfections of SD-RAN controllers in the control plane. To validate our results, we design `Delphi`, a new system architecture which is based on open-source code. `Delphi` provides theoretical models and measurement results for throughput predictions in SD-RAN environments, with respect to the maximum Channel Quality Indicator (maxCQI) and RR scheduling policies, respectively.

Based on the outcomes of this chapter, we are able to identify performance implications of SD-RAN controllers and provide analytical tractability. One of the main messages of this chapter is that for a maxCQI scheduler the data plane throughput is not affected considerably up to the point when the number of RAN elements in the data plane increases beyond 5000. Additionally, we show that the RR scheduler is completely insensitive to control packet losses.

To avoid potential controller undesired behavior that stems from a single controller approach, a distributed control plane is suggested. However, when considering a distributed control plane several issues among which the most important are, BS to controller mapping, transfer of BSs' databases and control synchronization, occur. In the case of real-time SD-RAN, the control handover overheads can potentially have a negative impact on control decisions and QoS of involved UE devices. However, this question remains unanswered in the literature. To overcome the aforementioned issues, in this chapter, we address the question of SD-RAN control plane design impact on the UE QoS performance. To achieve this, we develop a 5G simulator that contains the SD-RAN control plane messages and models controllers' behavior, based on measurements shown in Chapter 4. With the help of our simulator, we provide insights on distributed and centralized SD-RAN control plane effects on UE QoS considering metrics such as throughput, packet loss ratio and average packet delay.

Content and outline of this chapter: Section 5.1 provides related work on distributed SD-RAN control plane approaches as well as RAN virtualization. Section 5.2 presents `Delphi`, a novel system architecture which enables experimental evaluation and analytical tractability of throughput in SD-RAN environments, while leveraging available open-source code. In order to investigate the potential of a distributed SD-RAN control plane design, Section 5.3 presents to the best of our knowledge, the first 5G-enabled SD-RAN simulator that contains both single controller and distributed control plane designs and that is based on measurements with open-source SD-RAN controllers (`FlexRAN`). In this way, we can study the impact of SD-RAN control design on the UE QoS for a single controller and a distributed control plane, providing insights on control handover effects. Finally, Section 5.4, provides a summary of this chapter. The contents of this section are based on the following publications. The results with respect to `Delphi` are provided on [Pap+23a], whereas the 5G-enabled SD-RAN simulator is demonstrated in [Pap+22]

[Pap+23a] A. Papa, P. Kutsevol, F. Mehmeti, and W. Kellerer. "Effects of SD-RAN Control Plane Design on User Quality of Service." In: Proc. of the IEEE Conference on Network Softwarization (NetSoft). 2022. doi: 10.1109/NetSoft54395.2022. 9844029.

[Pap+22] A. Papa, P. Kutsevol, F. Mehmeti, and W. Kellerer. "Delphi: Computing the Maximum Achievable Throughput in SD-RAN Environments." In: IEEE Transactions on Network and Service Management (under revision). 2023.

5.1 Related Work

RAN has always played a major role in any generation of cellular networks. Its importance is even more emphasized in 5G, with its main task lying in provisioning of heterogeneous services in terms of their requirements and the establishment of programmable and flexible solutions compared to traditionally monolithic RAN infrastructures. The aforementioned stringent requirements hinder efficiency and collaboration between different RAN entities. A factor of paramount importance towards this transformation is related to the concepts of SD-RAN and RAN virtualization. Whereas related work on SD-RAN single controller scenarios was presented in Section 4.2, in this section attention is shifted towards literature review on distributed SD-RAN control plane.

5.1.1 RAN Virtualization

In the context of RAN virtualization, OpenAirInterface [Nik+14] and srsLTE [Gom+16] constitute the pillars of academic research. Both platforms provide a 3GPP-based softwarized platform for 4G/5G networks, where the core, RAN and UE components are running in pure software. Furthermore, they both allow for connection to SD-RAN controllers, where srsLTE is compliant with `5G-EmPOWER` [CKR19], whereas OpenAirInterface is compliant with both `FlexRAN` [Fou+16] and FlexRIC [SIN21].

Due to the high traction received in the last years, wireless platform emulators have been developed to allow for large-scale deployment of RAN environments [Bon+21a; Ber+19; Dan+19; Bre+21; Bon+21b]. Platforms such as Arena [Ber+19], POWDER [Bre+21] and Colosseum [Bon+21b] equip the community with powerful wireless emulators, with the latter being considered as the largest up-to-date. Additionally, Scope [Bon+21a], which is based on Colosseum [Bon+21b], provides a framework for open and softwarized prototypes.

Moreover, recent works such as CARES [Beg+18], vrAIn [Aya+20a], Concordia [FR21], and Nuberu [Gar+21] deal with resource allocation in such virtualized environments, providing valuable insights on the operation and management aspect. Yet, none of these works consider the large-scale experimentation with respect to SD-RAN environments, especially not with

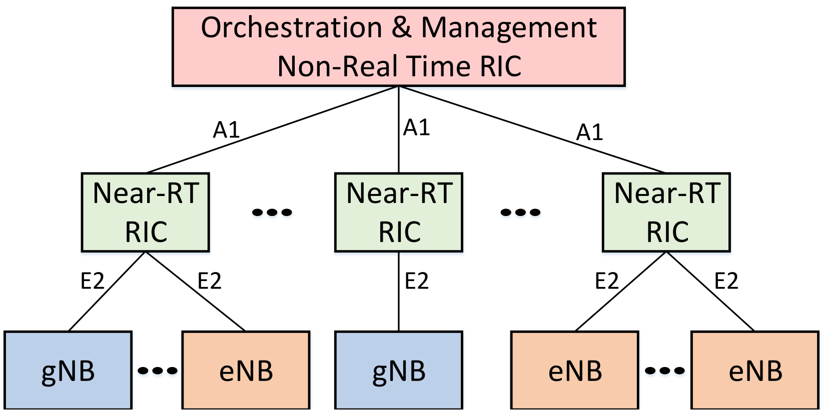


Figure 5.2: ORAN architecture overview. The control plane consists of a non-RT RIC responsible for management and orchestration. Multiple near-RT RICs tailored to time-critical applications such as scheduling and interference cancellation, enforce rules on BSs.

controllers under high load. For the latter scenario the authors do not provide results in terms of throughput evaluation, which is the main objective of this chapter.

5.1.2 SD-RAN Distributed Control Plane

The majority of works on SD-RAN envision a single controller in charge of the network [Yan+13; Moh+15; Gud+13; ASR15; Ros+17b]. Alternatively, [AWL15] and [Xu+16] suggest the introduction of a distributed control plane for SD-RAN, stating both the advantages in terms of scalability and enhanced control, while also demystifying the drawbacks in terms of induced operational complexity. But, they do not provide implementation details.

Recently, the ORAN alliance aims at standardizing the complex cellular system. In that regard, an SD-RAN platform and architecture concept have been developed, where the RAN Intelligent Controller (RIC) is introduced [O-Ra]. The control plane is composed of a non-RT RIC [O-Rc], responsible for the orchestration and management of one or multiple near-RT RICs [O-Re], mainly in charge of time-critical applications such as RAN scheduling, UE handover and interference management as portrayed in Fig. 5.2. ORAN also proposes the E2 protocol [O-Rd] for the communication of the near-RT RIC with the 4G BSs (i.e., Evolved NodeB (eNB)s) and 5G BSs (i.e., gNBs), and the A1 protocol for the interaction among non-real time and near-RT RICs.

However, while ORAN introduces the protocols and interfaces for the distributed SD-RAN control plane, there exists no information as of how the traffic balancing among the near-RT RICs and the control handover is performed. Moreover, implementations of such a distributed control plane are missing, leading to lack of insights on UE performance in those scenarios.

Furthermore, `FlexRAN` [Fou+16], `5G-EmPOWER` [CKR19] and Orion [FMK17], mainly address a single SD-RAN controller scenario, which may be insufficient when the number of RAN devices in the data plane increases drastically. Indeed, the results provided in Chapter 4 show that there is a significant performance deterioration for both `FlexRAN` and `5G-EmPOWER` when used in dense networks. In order to overcome the scalability issues stemming from a single controller, a distributed control plane architecture is suggested.

The closest implementation to a distributed-like architecture, considering the possibility of multiple SD-RAN controllers is FlexRIC [SIN21]. However, the focus of FlexRIC is mainly on the lean and lightweight control design and does not provide details with respect to the interaction of components in the distributed control plane, control handover or UE performance guarantees in such a scenario.

For our own design of a distributed SD-RAN control plane, we take into account approaches applied in the core network side [TG10; Xu+19; Dix+; Li+20a]. Yet, the consideration of the distributed control plane in SD-RAN architectures brings specific challenges. For instance, management applications executed by the SD-RAN controller depend on the highly variable wireless channel conditions. Thus, the controller has to collect and store the corresponding data, and this large amount of information has to be shared with the target controller during the control handover. Whereas the works devoted to control handover in core networks often neglect the overhead when transferring control-relevant information, this factor plays a crucial role in RAN. In this chapter, we design the control handover procedure, taking into account RAN specifications, which detaches our contribution from existing studies on distributed SDN.

5.2 Delphi: Towards Throughput Prediction in SD-RAN Environments

To predict the throughput of individual UEs as well as the total throughput (over all UEs) in SD-RAN environments, we present a novel system model, which we name `Delphi`. The name is inspired by the Greek mythology, where *Delphi* was home of Pythia, the oracle famous for predicting the future. Our approach is specifically engineered for 4G/5G systems that operate over virtualized RAN environments and are controlled by SD-RAN controllers with finite processing capacity. The system aims at characterizing throughput performance both mathematically and by measurements for the scenarios when SD-RAN controllers are overloaded. In this section, we elaborate on the main components of `Delphi`, and on their mutual interaction. Afterwards, we detail the scheduling policies utilized in our system and

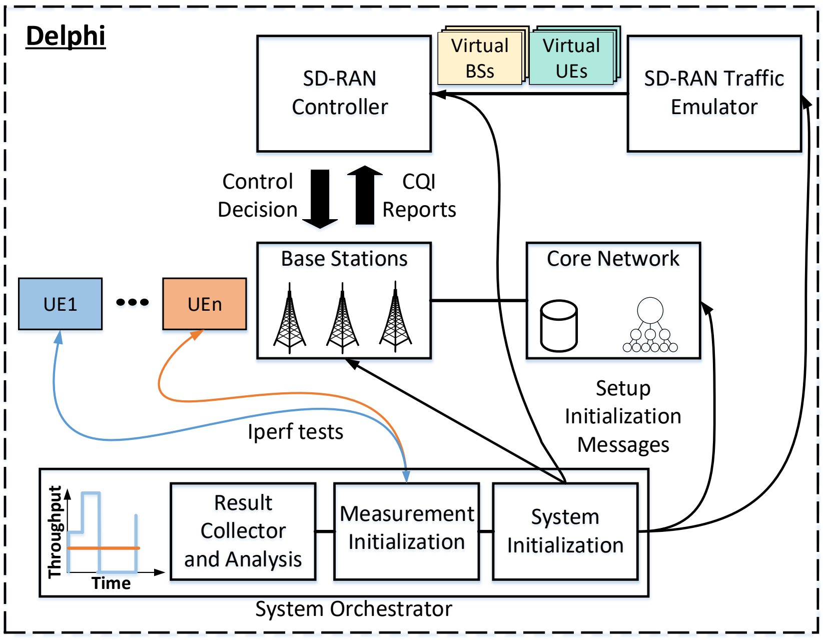


Figure 5.3: Overview of `Delphi`. The system is composed of the core network that assigns IPs to UEs, the SD-RAN controller and physical OpenAirInterface BSs. Additionally, the SD-RAN traffic emulator generates virtual UEs and BSs. The system orchestrator initializes the setup, initiates the measurements, collects the results and analyzes them.

the UE CQI patterns. Finally, we shed light on the control packet losses and the simulation framework.

5.2.1 System Design

The high-level architecture of `Delphi` is presented in Fig. 5.3. `Delphi` is based on open-source platforms and is in line with the 5G principles of programmability and softwarization.

As can be observed from Fig. 5.3, our system is composed of 5 main components: the core network, the BS (which in 5G is known as gNB) and UEs, the SD-RAN controller, the SD-RAN traffic emulator and the system orchestrator. A detailed explanation of all the components is presented as follows.

5.2.1.1 Core Network

The core network is based on the 4G 3GPP standard compliant OpenAirInteface core [Eur] and it mainly provides 3 functionalities. These functionalities consist of the HSS which stores the database of UEs that access the network, the MME that keeps track of the UE mobility and the SGW/PGW which are responsible for assigning IPs to UEs in order to maintain a connection. The reason for choosing the 4G core version of OpenAirInterface is related to its stability compared to the 5G counterpart. Moreover, the choice of the core network does not alter the performance of scheduling in RAN, which is within the scope of this thesis.

5.2.1.2 SD-RAN Controller

The SD-RAN controller is based on the `FlexRAN` principle [Fou+16], but utilizes the latest code from Eurecom [Mosa]. In our system, the control of the network, in particular with respect to MAC scheduling, involves the SD-RAN controller. That entails the scheduling to be performed in two steps. Initially, the SD-RAN controller distributes wireless resources, referred to as PRBs, among the BSs in what is called **control decision** based on a certain **control scheduling policy**. In turn, the BSs distribute the assigned PRBs from the SD-RAN controller among the corresponding UEs using the traditional **MAC scheduling**. `Delphi` utilizes the `FlexRAN` API [Mos20] and `FlexRAN` protocol [Fou+16] for the communication with the BS. Every BS has a unique connection to the `FlexRAN` controller via its own link, and the correct delivery is ensured by means of TCP.

The control decision is sent from the SD-RAN controller towards the BSs every 1 s, which we call **control period**. This decision is based on channel statistics containing information such as the CQI of UEs within BSs. In this case, the 1 s period is tied to the granularity of *iperf* measurements. To enable this procedure, `Delphi` extends the available `FlexRAN` code and implements a set of control scheduling policies, as detailed in Section 5.2.2.

5.2.1.3 Base Station

The BS is the main component of any RAN system. Within the BS, several UEs are assigned wireless resources depending on their CQIs and the MAC scheduler operating at the BS. In this section, for the BS and UEs we utilize the emulation mode of OpenAirInterface based on the `mosaic5g-oai-sim` branch [Mosc]. The rationale for this choice is twofold. Firstly, the `mosaic5g-oai-sim` provides a realistic wireless channel model based on 3GPP standardization. Secondly, the latter enables a manual configuration and control of CQI values for UEs. That leads to an easier deployment and assessment of results utilizing realistic UE traces, but also fosters reproducibility and allows analytical tractability.

5.2.1.4 SD-RAN Traffic Emulator

The SD-RAN traffic emulator is based on `MARC` [Pap21] presented in Chapter 4. This tool generates virtual BSs and virtual UEs that follow the `FlexRAN` protocol. `Delphi` utilizes the traffic emulator in order to generate background traffic in the network and stress the SD-RAN controllers to their capacity limits in order to track their capabilities. Here we emphasize that

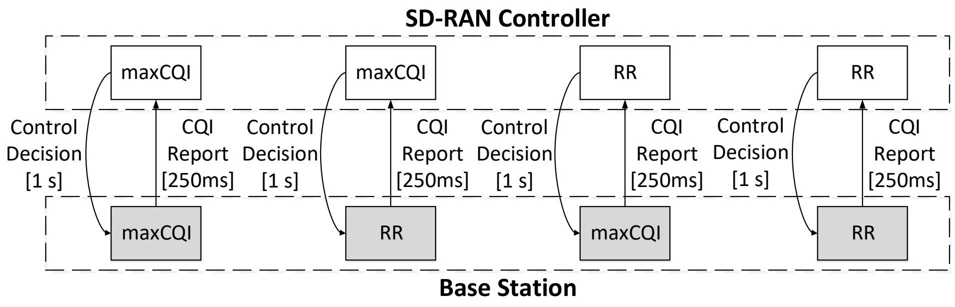


Figure 5.4: Overview of the possible system policies. The scheduling is performed in two steps where the control scheduling takes places at the SD-RAN controller and the MAC scheduling at the BS.

for the virtual BSs and UEs generated from `MARC` we cannot measure their throughput due to the fact that they do not connect to the core network. Hence, they do not receive IPs. This is only available for the OpenAirInterface BSs.

5.2.1.5 System Orchestrator

The orchestrator block of `Delphi` is based on scripts that are used to configure network components, such as BSs, core network and the SD-RAN controller. Moreover, the orchestrator initiates various measurements and finally collects and analyzes the results. In principle, we follow the logic of Mosaic5G [Mosb]. Yet, we adapt the respective scripts to achieve the automation of: 1) initialization of all system components, 2) adaptation of UE CQI patterns based on a case study and realistic traces, 3) seamless communication with the SD-RAN controller, and 4) throughput measurements, result collection and analysis.

5.2.2 Control and MAC Scheduling Policies

As mentioned previously, the scheduling in `Delphi` is implemented in two steps. Initially, it is performed at the SD-RAN controller (i.e., control scheduling) and then at the BS (i.e., MAC scheduling). The SD-RAN controller updates the resource distribution to BSs every 1 s through the control decision. These updates are based on CQI reports from the BSs with respect to their UEs, as illustrated in Fig. 5.4.

Different scheduling policies running at the SD-RAN controller and within the BS produce different results in terms of the throughput. For instance, if the RR policy is used at the SD-RAN controller, then BSs within the controller's operational region share equally the wireless resources among them. In turn, UEs within the BSs will have to share the received wireless resources too. Whereas this may result in a fair distribution of resources among UEs, it will not produce the highest achievable throughput.

On the other hand, if the maxCQI scheduling policy is used within the SD-RAN controller, the BS whose UEs achieve the highest CQI during the control period, obtains all the resources. In turn, the UEs within the BS with the highest CQI receive all the resources of that BS assuming that the UEs always have data to send. While this approach produces the highest throughput in the network, it may lead to throughput starvation for some UEs (those which experience bad channel conditions). That happens due to the fact that UEs with low CQI values almost never receive resources. Covering two of the most important aspects of a cellular network, which are overall throughput maximization and providing robust behavior under system imperfections is the reason for choosing the two aforementioned policies (maxCQI and RR) for the analysis presented in this chapter.

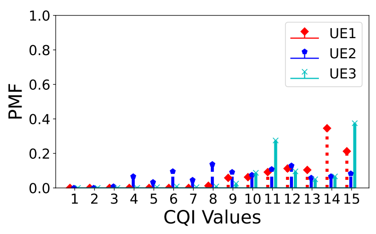
While deciding on the appropriate scheduler within the BS is hard, adding the entity of the SD-RAN controller in the scheduling decision renders the system even more complex, since now there are many combinations of scheduling policies to decide on. In this chapter, we demystify results for two of the scheduling policies depicted in Fig. 5.4, namely the maxCQI-maxCQI and RR-RR scheduler. The first term refers to the control scheduling policy, whereas the second to the MAC scheduling policy.

5.2.3 CQI Patterns

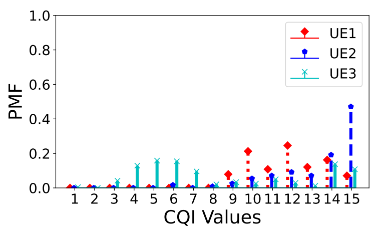
In a cellular system, the UE throughput is impacted by its CQI value. This can be observed in Fig. 5.6, where the results were obtained with `Delphi`. The x-axis represents the range of UE CQI values, while the y-axis the achieved throughput in the system when all 25 PRBs are assigned to that UE. The minimum with all the resources assigned to the same UE is achieved when $CQI = 1$, and is 0.62 Mbps, whereas the maximum is achieved for $CQI = 15$ - in total 18.3 Mbps.

Given that CQI values have a direct impact on the achievable UE throughput, in this subsection we shed light on the selection of the CQI values for our evaluations. To that end, we consider two scenarios:

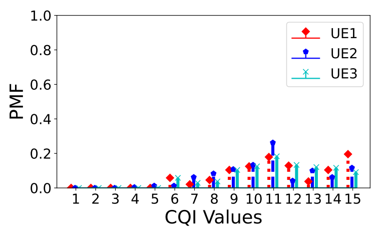
- A **case study**, where UEs have either the CQI of 15 (representing excellent channel conditions) or 7 (representing medium channel conditions). There is an equal probability for a UE to have either one of these CQIs.
- **Real traces** for UEs, as depicted in Fig. 5.5, which are obtained from [Rac+20] and [MP21], where for 3 BSs we depict CQI patterns of various UEs, all with different Probability Mass Function (PMF).



(a) PMF of CQI values for 3 UEs, BS1.



(b) PMF of CQI values for 3 UEs, BS2.



(c) PMF of CQI values for 3 UEs, BS3.

Figure 5.5: PMF for 3 BSs and 3 UEs within each BS based on realistic UE traces from a 5G dataset [Rac+20].

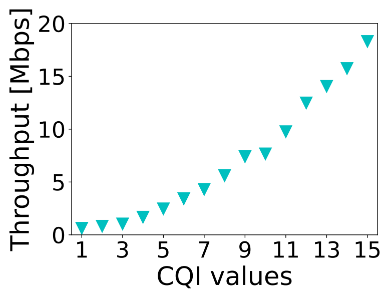


Figure 5.6: CQI to throughput conversion for 25 PRBs with measurements provided by `Delphi`.

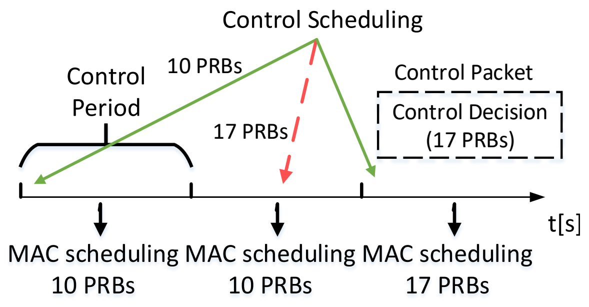


Figure 5.7: Scenario of control packet losses. The MAC scheduling is updated every control period. Lost packets in dashed red, received packets shown in green.

5.2.4 Control Packet Losses

The first scheduling step happens at the SD-RAN controller every 1 s (control period) and the decision (with the allocated resources) is sent to the BS. Once the BS receives the decision with the respective PRB allocation, it updates the previously stored one. However, when the load of the SD-RAN controller increases, due to the limited processing capacity of the controller, some of the control decisions sent to the BSs arrive with delay. As demonstrated in Fig. 5.1, when the number of RAN elements reaches 10000, the aggregated load at the controller reaches 250 Mbps, overloading it and making it prone to operating with flaws. We note that these imperfections in controller behavior are not caused by the TCP, since as portrayed in Fig. 5.1, for 1 BS towards the controller even for 100 UEs the necessary connection capacity does not surpass 5 Mbps. In our system, we define two use cases. Initially, we assume that when a control packet arrives at the BS with delays, it is discarded and considered as **lost**. In that case, the control decision is not updated and the BS keeps the previous one. This can lead to throughput degradation as the channel conditions may have changed drastically from the previous control period.

An example of this scenario is depicted in Fig. 5.7. Every control period i.e., 1 s, the BS receives the control decision, containing the new resource allocation from the SD-RAN controller, and updates its last stored resource allocation scheme from the previous control period. When the packet is received timely, then the control packet reaches the BS, which is depicted with green. Otherwise, the control packet is considered as lost, as it is shown in red dashed line in Fig. 5.7.

While the first scenario is easier in terms of analytical tractability, in reality the control packet is not lost, but it reaches the BS with a delay. Therefore, in our second use case, we drop this assumption and measure directly the effect of the delayed control packet on the UE throughput with respect to background traffic generated by virtual BSs and UEs, as detailed in Section 5.2.7.

5.2.5 Simulation Framework

In order to verify the measurements and the theoretical model provided by `Delphi`, we also provide a simulation framework that serves as our **baseline**. The simulation framework contains the actual information with respect to UE CQI values in all scenarios. The conversion of CQI to throughput is applied according to the measurements provided by `Delphi`, as shown in Fig. 5.6. For every UE in each BS, a list is created, where each entry corresponds to a measurement point that represents a control period. Each list entry contains a CQI value following the distributions depicted in Fig. 5.5, or the case study (CQIs of 7 and 15 with identical probabilities of 1/2). Possessing knowledge with respect to all CQIs at each measurement point (control period), an assessment occurs to compare the CQIs among all UEs and BSs. Depending on the scheduling policy (see Fig. 5.4), the resources are distributed to BSs and consequently to UEs. The achieved throughput then depends on the CQI in the list entry, and the scheduling policy.

Regarding the control packet losses, at each measurement point, the control packet is either lost or received timely at the BS as illustrated in Fig. 5.7. At the beginning of each simulation, according to the assumed ratio of lost control packets, a list with binary entries is generated, containing 0 if the control packet is received timely and 1 if it is lost. This generation occurs randomly. If the control packet is lost, then the PRB allocation from the previous measurement point is applied, otherwise the exact resource allocation is carried out.

5.2.6 Theoretical Model

As elaborated in Section 2.1.3.2, similar to 4G, the block resource allocation scheme is used in 5G as well, with *PRB* being the allocation unit [KW15]. But allowing higher flexibility in choosing the subcarrier spacing, and consecutively, the PRB bandwidth. This then conditions the duration of the slot. Within a slot, different PRBs are assigned to different UEs. In general, the assignment varies across slots. Consequently, scheduling is to be performed across two dimensions, *frequency* and *time*. The total number of PRBs in the system is K.

In general, UEs experience different channel conditions in different PRBs even within the same slot, and therefore a different per-block SINR. Because of UE mobility and time-varying channel characteristics, per-PRB SINR changes from one slot to another. This changing *per-PRB* SINR translates into a varying per-PRB rate. The value of SINR in a slot determines the CQI (a parameter sent by the UE to the BS), which depending on the Modulation and Coding Scheme (MCS) sets the per-PRB rate. There are 15 possible values of CQI [ETS18]. E.g., if at time t the per-PRB SINR lies in the interval $[\gamma_l, \gamma_{l+1}]$, with γ_l and γ_{l+1} being the thresholds of the CQI ($l = 1, \ldots, 15$), the per-PRB rate would be $r_l(t)$ [MR19]. To maintain analytical tractability and compare the results under the same conditions with OpenAirInterface, in this section we make a simplifying assumption. Specifically, we assume that the BS splits the transmission power equally among all the K PRBs, and that the channel characteristics for a UE remain unchanged across all PRBs (i.e., identical CQI over all PRBs), but they change randomly from one slot to another. This reduces the resource allocation problem to the number of allocated PRBs and not to which PRBs are assigned to a given UE.

Having in mind the previous assumptions, it follows that in every slot the per-PRB rate of UE j can be modeled as a discrete random variable, R_j, with values in $\{r_1, r_2, \ldots, r_{15}\}$, such that $r_1 < r_2 < \ldots < r_{15}$, with a PMF $p_{R_j}(x)$. The latter is a function of UE's j SINR over time. For notational simplicity, we omit the reference to time from now on. In this section, we focus our attention on two scheduling policies - maxCQI, where the best UE or BS gets all the PRBs and RR, where resources are split equally among all the entities. When two or more entities have identical (and the best) channel conditions in the control period, the resources are split equally among them with the maxCQI scheduler. Different scheduling policies can be implemented on the controller and on BSs. In the first case, we present the result when the RR scheduling is implemented at both entities. Then, we focus on maxCQI.

5.2.6.1 Round-Robin Scheduling Policy

When using this scheduling policy both at the controller and at the BSs, we assume that the PRBs are shared equally among all the BSs, and within the coverage area of a given BS, the PRBs are shared among all its active UEs.

In the general case of M BSs, where the number of UEs in each BS is denoted by $n_i, \forall i \in \{1, \ldots, M\}$, every BS would receive $\frac{K}{M}$ PRBs. We use $R_{i,j}$ to denote the per-PRB rate that UE j receives from BS i. The average throughout of UE j, being served by BS i, is

$$\mathbb{E}[Th_{i,j}] = \frac{K}{Mn_i}\mathbb{E}[R_{i,j}]. \tag{5.1}$$

The throughput in BS i is then

$$\mathbb{E}[Th_i] = \sum_{j=1}^{n_i}\mathbb{E}[Th_{i,j}]. \tag{5.2}$$

Therefore, we have the following:

Result 1 *The total throughput in the system is*

$$\mathbb{E}[Th]=\sum_{i=1}^{M}\mathbb{E}[Th_i]=\sum_{i}^{M}\sum_{j=1}^{n_i}\mathbb{E}[Th_{i,j}] = \frac{K}{M}\sum_{i=1}^{M}\frac{1}{n_i}\sum_{j=1}^{n_i}\mathbb{E}[R_{i,j}]. \tag{5.3}$$

Note that the RR scheduling policy is completely insensitive to the loss of control packets; the assignment of resources is fixed at all times.

5.2.6.2 maxCQI Scheduling Policy

To ease the presentation, in this section, we consider the case with two BSs and with a single UE in each of them. Then, we explain how the analysis can be extended to multiple UEs per BS. Let R_1 denote the per-PRB rate of the UE in BS 1. As already mentioned, it can take a value from the discrete set $\{r_1, \ldots, r_{15}\}$, depending on the value of CQI in a slot. Similarly, R_2 denotes the per-PRB rate of the UE in BS 2. We denote the PMF of R_1 and R_2 by $p_{R_1}(x)$ and $p_{R_2}(x)$, respectively.

Let q denote the probability of the control packet loss. This can arise as a consequence of a large number of UEs and BSs sending their packets to the controller with finite processing capacity. A packet not processed by the end of the control period at the controller is considered as lost at the BS and in that case the scheduling decisions from the previous control period are enforced (see Fig. 5.7). First, we solve the scenario when $q = 0$, i.e., in the case when the controller processes and allocates the resources timely.

We also need this result for the case when there are packets that are lost, too. Conditioning upon the possible outcomes, we obtain the average throughput as

$$\mathbb{E}[Th_{q=0}] = \mathbb{P}(R_1 > R_2)K\mathbb{E}[R_1|R_1 > R_2] + \mathbb{P}(R_1 = R_2)\frac{K}{2}\mathbb{E}\left[R_1 + R_2|R_1 = R_2\right] + \mathbb{P}(R_1 < R_2)K\mathbb{E}\left[R_2|R_1 < R_2\right]. \tag{5.4}$$

Namely, in a slot, either R_1 and R_2 are identical, or one of them is higher than the other. For instance, when $R_1 > R_2$, the average throughput in that slot is $K\mathbb{E}[R_1|R_1 > R_2]$, because the UE in BS 1 gets all the resources. A similar allocation is performed when the UE in BS 2 has better channel conditions. If both UEs have identical R, they receive the same amount of PRBs, i.e., $\frac{K}{2}$, as can be observed from Eq.(5.4). We derive next all the terms from Eq.(5.4). The first probability term yields

$$\mathbb{P}(R_1 > R_2) = \sum_{x=r_1}^{r_{15}} \mathbb{P}(R_1 > R_2|R_2 = x)p_{R_2}(x) = \sum_{x=r_1}^{r_{15}} \bar{F}_{R_1}(x)p_{R_2}(x), \tag{5.5}$$

where $\bar{F}_{R_1}(x) = 1 - \mathbb{P}(R_1 \leq x)$ is the Complementary Cumulative Distribution Function (CCDF). Similarly, for the other two probability terms, we have

$$\mathbb{P}(R_1 = R_2) = \sum_{x=r_1}^{r_{15}} p_{R_1}(x)p_{R_2}(x), \text{and} \tag{5.6}$$

$$\mathbb{P}(R_1 < R_2) = \sum_{x=r_1}^{r_{15}} \bar{F}_{R_2}(x)p_{R_1}(x). \tag{5.7}$$

For the final result of the first expectation term in Eq.(5.4), after some algebra, we obtain

$$\mathbb{E}[R_1|R_1 > R_2] = \frac{\sum_{x=r_1}^{r_{15}} \sum_{y=r_1}^{x-\epsilon} x \cdot p_{R_1}(x)p_{R_2}(y)}{\sum_{x=r_1}^{r_{15}} \bar{F}_{R_1}(x)p_{R_2}(x)}. \tag{5.8}$$

Note that the denominator in Eq.(5.8) is simply $\mathbb{P}(R_1 > R_2)$ which is already derived in Eq.(5.5). Also, observe that since R_1 and R_2 can take values from a finite discrete set, and the expectation in Eq.(5.8) is conditioned upon the strict inequality between R_1 and R_2, we use the variable $\epsilon > 0$ to denote a very small positive quantity near 0. This implies that for a given value of R_1, R_2 has to be lower than that. Similarly, for the third expectation term in Eq.(5.4) we get

$$\mathbb{E}[R_2|R_1 < R_2] = \frac{\sum_{x=r_1}^{r_{15}} \sum_{y=r_1}^{x-\epsilon} x \cdot p_{R_2}(x)p_{R_1}(y)}{\sum_{x=r_1}^{r_{15}} \bar{F}_{R_2}(x)p_{R_1}(x)}, \tag{5.9}$$

whereas for the second expectation term in Eq.(5.4), we obtain

$$\mathbb{E}[R_1 + R_2|R_1 = R_2] = \frac{\sum_{x=r_1}^{r_{15}} 2 \cdot x \cdot p_{R_1}(x)p_{R_2}(x)}{\sum_{x=r_1}^{r_{15}} p_{R_1}(x)p_{R_2}(x)}. \tag{5.10}$$

5.2.6.3 Packet losses

In this case, control packets are received at the BS with a delay, i.e., $q > 0$. Whenever a packet is delayed, it is dropped and considered lost. In that case, PRBs are allocated in the same way as in the previous control period. With this logic, we have the following:

Result 2 *The average throughput when there are packets that are lost is expressed as*

$$\mathbb{E}[Th] = (1 - q)\mathbb{E}[Th_{q=0}] + q\mathbb{E}[Th_{q>0}]. \tag{5.11}$$

The term $\mathbb{E}[Th_{q=0}]$ in Eq.(5.11) corresponds to the instants when the packet is received timely, which occurs with probability $1 - q$. Essentially, it is equal to Eq.(5.4). The second right-hand side term, $\mathbb{E}[Th_{q>0}]$, captures the throughput when the packet is lost, which occurs with probability q. It is expressed as

$$\begin{gathered}\mathbb{E}[Th_{q>0}] = \mathbb{P}(R_1 < R_2)\mathbb{E}[A|R_1 < R_2]+ \\ \mathbb{P}(R_1 = R_2)\mathbb{E}[B|R_1 = R_2] + \mathbb{P}(R_1 > R_2)\mathbb{E}[C|R_1 > R_2].\end{gathered} \tag{5.12}$$

It is worth mentioning that in Eq.(5.12) we condition the expectation on three events in the last control period before the packet was lost: whether the UE in BS 1 had a better channel, whether both UEs had identical channels, or whether the channel of the UE in BS 2 was better during the control period prior to the control period the packet was lost.

The average system throughput when the UE in BS 2 had a better channel in the previous control period when the packet was received timely, provided that in the current control period the packet is lost, $\mathbb{E}[A|R_1 < R_2]$, is given by

$$\begin{gathered}\mathbb{E}[A|R_1 < R_2] = \mathbb{P}(R_1 < R_2)\mathbb{E}[KR_2|R_1 < R_2]+ \\ \mathbb{P}(R_1 = R_2)\mathbb{E}[KR_2|R_1 = R_2] + \mathbb{P}(R_1 > R_2)\mathbb{E}[KR_2|R_1 > R_2].\end{gathered} \tag{5.13}$$

Let us look more closely at the individual terms of Eq.(5.13). If in the current control period (when the packet is lost), $R_2 > R_1$, the average throughput obtained is $\mathbb{E}[KR_2|R_1 < R_2]$. Namely, again the UE of BS 2 receives all the resources as it was the best UE in the previous control period. So, in this case the policy assigns the PRBs correctly despite the fact that the packet was lost. On the other hand, if the UE channels are identical or the UE of BS 1 has a better channel in the control period with lost packet, still the UE of BS 2 obtains all the resources, i.e., $\mathbb{E}[KR_2|R_1 = R_2]$ and $\mathbb{E}[KR_2|R_1 > R_2]$, respectively, in spite of not having the highest CQI anymore.

Following a similar reasoning, when both UEs had identical channels in the previous control period, and in the current control period the packet is lost, the average throughput is

$$\begin{aligned}\mathbb{E}[B|R_1 = R_2] = \mathbb{P}(R_1 < R_2)\mathbb{E}\left[\frac{KR_1}{2} + \frac{KR_2}{2}|R_1 < R_2\right] + \\ \mathbb{P}(R_1 = R_2)\mathbb{E}\left[\frac{KR_1}{2} + \frac{KR_2}{2}|R_1 = R_2\right] + \\ \mathbb{P}(R_1 > R_2)\mathbb{E}\left[\frac{KR_1}{2} + \frac{KR_2}{2}|R_1 > R_2\right]. \end{aligned} \tag{5.14}$$

Finally, if the UE of BS 1 had a better channel in the previous control period, the corresponding average throughput is

$$\begin{aligned}\mathbb{E}[C|R_1 > R_2] = \mathbb{P}(R_1 < R_2)\mathbb{E}[KR_1|R_1 < R_2] + \\ \mathbb{P}(R_1 = R_2)\mathbb{E}[KR_1|R_1 = R_2] + \mathbb{P}(R_1 > R_2)\mathbb{E}[KR_1|R_1 > R_2],\end{aligned} \tag{5.15}$$

meaning that the UE of BS 1 will receive all the resources when a packet is lost.

In Eqs.(5.12)-(5.15), the expressions for $\mathbb{P}(R_1 < R_2)$, $\mathbb{P}(R_1 > R_2)$, and $\mathbb{P}(R_1 = R_2)$ were already derived in Eqs.(5.5)-(5.7). Eqs.(5.8)-(5.10) can be used for some of the terms in Eqs.(5.12)-(5.15). Finally, for the remaining terms, we have

$$\mathbb{E}[R_1|R_1 = R_2] = \mathbb{E}[R_2|R_1 = R_2] = \frac{\sum_{x=r_1}^{r_{15}} x \cdot p_{R_1}(x) p_{R_2}(x)}{\sum_{x=r_1}^{r_{15}} p_{R_1}(x) p_{R_2}(x)}. \tag{5.16}$$

We substitute Eqs.(5.5)-(5.10), (5.16) into Eqs.(5.13)-(5.15), and furthermore we replace the so obtained Eqs.(5.13)-(5.15) into Eq.(5.12). Then, we substitute the latter together with Eq.(5.4) into Eq.(5.11) to obtain the total system throughput.

In this section, we have shown the derivation of the overall throughput. The procedure for obtaining the throughput of individual UEs is straightforward. Let us consider the UE in BS 1. The differences are as follows. Eq.(5.4) contains only the first two terms (no allocated resources to that UE when the other BS has better channel). For the same reason, Eq.(5.12) contains only the second and third term. The remainder of the procedure (all the other expressions from Section 5.2.6.2, including Eq.(5.11)) remains unchanged. Similar procedure is followed for the UE of BS 2.

5.2.6.4 Scenario: 3 BSs, no Packet Losses

Next, we consider the scenario with three BSs, and one UE in each of them for the maxCQI policy. Only the analysis for $q = 0$ is shown, whereas we omit the analysis for $q > 0$ due to no further technical interest. Following a similar reasoning as with 2 BSs, we have:

Result 3 *The average throughput with three BSs, where in the area of each one of them there is one UE with per-PRB rates R_1, R_2, and R_3 is*

$$\begin{aligned}\mathbb{E}[Th_{q=0}] = \mathbb{P}(R_1 > R_2, R_1 > R_3)K\mathbb{E}[R_1|R_1 > R_2, R_1 > R_3]\\ +\mathbb{P}(R_1 = R_2, R_1 > R_3)\frac{K}{2}\mathbb{E}[R_1+R_2|R_1 = R_2, R_1 > R_3]\\ +\mathbb{P}(R_1 = R_3, R_1 > R_2)\frac{K}{2}\mathbb{E}[R_1+R_3|R_1 = R_3, R_1 > R_2]\\ +\mathbb{P}(R_1 = R_2, R_1 = R_3)\frac{K}{3}\mathbb{E}[R_1+R_2+R_3|R_1 = R_2, R_1 = R_3]\\ +\mathbb{P}(R_2 > R_1, R_2 > R_3)K\mathbb{E}[R_2|R_2 > R_1, R_2 > R_3]\\ +\mathbb{P}(R_2 = R_3, R_2 > R_1)\frac{K}{2}\mathbb{E}[R_2+R_3|R_2 = R_3, R_2 > R_1]\\ +\mathbb{P}(R_3 > R_1, R_3 > R_2)K\mathbb{E}[R_3|R_3 > R_1, R_3 > R_2].\end{aligned} \tag{5.17}$$

Note that in Eq.(5.17), a single BS gets all the K PRBs when its UE has the best channel. When two of the BS's UEs have identical per-PRB rate in a control period and the third BS's UE worse channel, those two BSs split the available K PRBs equally. Finally, if the three UEs have the same per-PRB rate in the control period, each one of them will receive $\frac{K}{3}$ of the PRBs in that control period. For the probability terms in Eq.(5.17) we can write:

$$\mathbb{P}(R_i > R_j, R_i > R_k) = \sum_{x=r_1}^{r_{15}}\sum_{y=r_1}^{x-\epsilon}\sum_{z=r_1}^{x-\epsilon} p_{R_i}(x)p_{R_j}(y)p_{R_k}(z), \tag{5.18}$$

$$\mathbb{P}(R_i = R_j, R_i > R_k) = \sum_{x=r_1}^{r_{15}}\sum_{z=r_1}^{x-\epsilon} p_{R_i}(x)p_{R_j}(x)p_{R_k}(z), \tag{5.19}$$

for $i=\{1,2,3\}$, $j=\{1,2,3\}$, $k=\{1,2,3\}$, and $i\neq j\neq k$. Also,

$$\mathbb{P}(R_1 = R_2, R_1 = R_3) = \sum_{x=r_1}^{r_{15}} p_{R_1}(x)p_{R_2}(x)p_{R_3}(x). \tag{5.20}$$

For the expectation terms in Eq.(5.17), following the basic rules from the theory of probability, we have:

$$\mathbb{E}[R_i|R_i > R_j, R_i > R_k] = \frac{\sum_{x=r_1}^{r_{15}}\sum_{y=r_1}^{x-\epsilon}\sum_{z=r_1}^{x-\epsilon} x p_{R_i}(x)p_{R_j}(y)p_{R_k}(z)}{\sum_{x=r_1}^{r_{15}}\sum_{y=r_1}^{x-\epsilon}\sum_{z=r_1}^{x-\epsilon} p_{R_i}(x)p_{R_j}(y)p_{R_k}(z)}, \tag{5.21}$$

$$\mathbb{E}[R_i+R_j|R_i = R_j, R_i > R_k] = \frac{\sum_{x=r_1}^{r_{15}}\sum_{z=r_1}^{x-\epsilon} 2x p_{R_i}(x)p_{R_j}(x)p_{R_k}(z)}{\sum_{x=r_1}^{r_{15}}\sum_{z=r_1}^{x-\epsilon} p_{R_i}(x)p_{R_j}(x)p_{R_k}(z)}, \tag{5.22}$$

for $i=\{1,2,3\}$, $j=\{1,2,3\}$, $k=\{1,2,3\}$, and $i\neq j\neq k$. Similarly,

$$\mathbb{E}[R_1+R_2+R_3|R_1 = R_2, R_1 = R_3] = \frac{\sum_{x=r_1}^{r_{15}} 3x p_{R_1}(x)p_{R_2}(x)p_{R_3}(x)}{\sum_{x=r_1}^{r_{15}} p_{R_1}(x)p_{R_2}(x)p_{R_3}(x)}. \tag{5.23}$$

A similar procedure, only with more expectation and probability terms, is followed in the case with more than three BSs. The respective analysis and detailed results of the latter can be found in our work [MPK23].

5.2.6.5 maxCQI-maxCQI: Multiple UEs per BS

While the previous analysis was concerned only with one UE per BS, the generalization to any number of UEs where maxCQI scheduling policy is implemented on both the BS level and UE level is almost straightforward. Namely, in the analysis above, we only need to replace, e.g., R_1, R_2, and R_3 with $\max R_1$, $\max R_2$, and $\max R_3$, respectively, where $\max R_i, i = \{1, 2, 3\}$, is the UE with the highest CQI within BS i. Then, using some algebra, the final result follows easily.

5.2.6.6 Other Scheduling Policies

For the maxCQI-RR policy, e.g., in the case of two BSs, the first BS in a control period will receive all the PRBs with probability $\mathbb{P}(\max R_1 > \max R_2)$, in which case each UE within that BS will receive K/n_1 PRBs, where n_1 is the number of active UEs in BS 1. BS 2 will receive all the K PRBs with probability $\mathbb{P}(\max R_1 < \max R_2)$, where each UE within BS 2 will receive K/n_2 PRBs, with n_2 being the number of active UEs in BS 2. If $\max R_1 = \max R_2$, then each UE in BS 1 will receive $K/2n_1$ PRBs, whereas each UE in BS 2, $K/2n_2$ PRBs. These facts are then combined in an analysis similar to that performed in Section 5.2.6.2.

For the RR-maxCQI policy, the system is decomposed into multiple independent BSs (with a fixed and equal number of PRBs), for which the analysis in Section 5.2.6.2 holds.

5.2.7 Performance Evaluation

In this section, we evaluate the performance of our system with respect to the effect of SD-RAN control plane imperfections in the data plane. We split our results in two parts. In the first part, we assume a certain control packet loss ratio in the control plane between the SD-RAN controller and the BS regarding the control decision. The second part drops this assumption and evaluates the controller performance given additional background traffic. This is achieved by increasing the amount of BSs and UEs that the controller has to manage.

Initially, we portray an example of the control packet loss for 2 BSs considering a maxCQI-maxCQI scheduler. Further, we investigate the drop in the sum throughput that is accompanied by the time instance of packet loss. Moreover, we demonstrate outcomes for two use cases, namely the case study and the real traces (Section 5.2.3). Evaluations are demonstrated for

2 and 3 OpenAirInterface BSs for maxCQI-maxCQI and RR-RR scheduling policies. For 2 BSs, we verify our measurements with simulations and theoretical results for all the assumed control packet loss ratios. On the other hand, for 3 BSs, we provide the theoretical result for the case where no packets are lost. Results are demonstrated in boxplots. For the measurements, each configuration is repeated 10× for 270 s, by removing the initial and last 15 s to eliminate the transient phase of the measurements. In turn, the simulations are performed 100×.

When the packet loss assumption is dropped, the depicted results for 2 and 3 BSs are generated with OpenAirinterface (Section 5.2.1.3), while considering additional emulated UEs and BSs (Section 5.2.1.4). In that regard, we show how the network load affects the throughput drop in % for maxCQI-maxCQI and achieved throughput for the RR-RR scheduling policies, as the latter is insensitive to control packet losses.

5.2.7.1 Measurement Setup

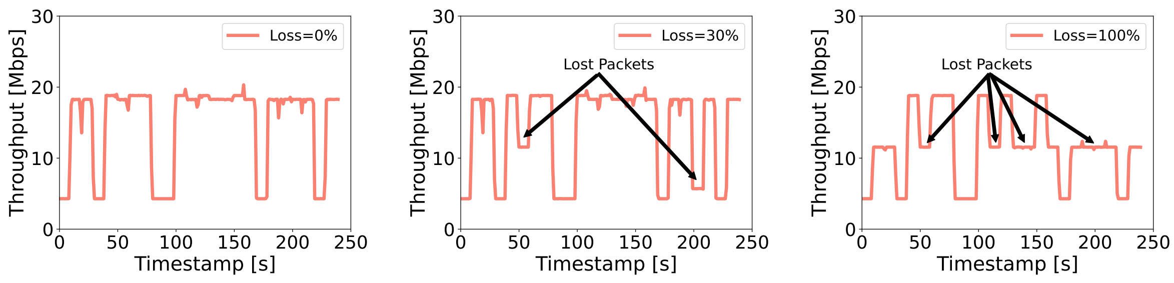
The measurement setup follows the system design from Fig. 5.3, where 1 Desktop PC is running the SD-RAN benchmark [Pap21] presented in Section 4. The PC is equipped with an Intel(R) Core(TM) i5-3470 CPU, with 4 CPU cores @ 3.2 GHz and 8 GB RAM running Ubuntu 18.04.2 LTS with 4.15.0-58-generic kernel. For the `FlexRAN` SD-RAN controller, a server running an Intel(R) Xeon(R) CPU E5-2650 v4 @ 2.20 GHz is set. The server contains 12 physical CPU cores and 64 GB of RAM operating under Ubuntu 18.04.3 LTS with 4.15.0-66-generic kernel.

Furthermore, OpenAirInterface BSs and the core network operate in an Intel(R) Core(TM) i7-7700T CPU @ 2.9 GHz. Each BS PC contains 4 physical CPU cores and 16 GB of RAM, whereas the core network, since it is not that demanding, only contains 1 CPU and 4 GB of RAM. The operating system is Ubuntu 16.04.04 LTS with 4.4.0-116-generic kernel.

Unless stated otherwise, a scheduling application is running on the SD-RAN controller server, which collects CQI values sent by the BSs every 250 ms. This is a design choice, which is picked smaller than 1 s (control period) in order to provide enough measurements for the controller to achieve optimal resource allocation. The CQI values of the UEs within the BSs also change every 1 s [Rac+20; MP21] and their PMFs for 2 and 3 BSs are illustrated in Fig. 5.5.

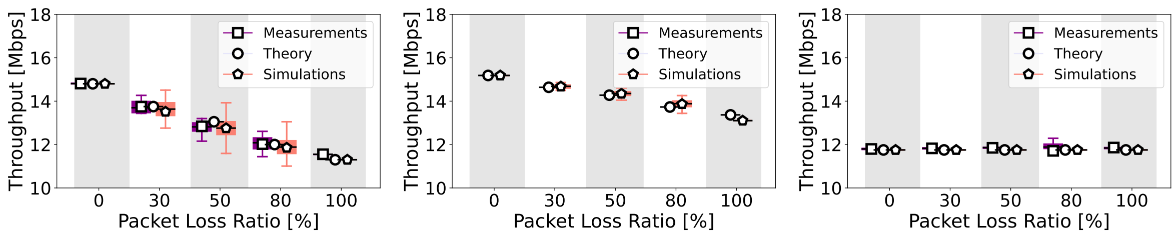
5.2.7.2 Control Packet Loss Ratio Assumption

Here, we assume that a certain amount of packets in the control plane are lost. These messages contain control decisions with respect to the resource allocation sent from the SD-



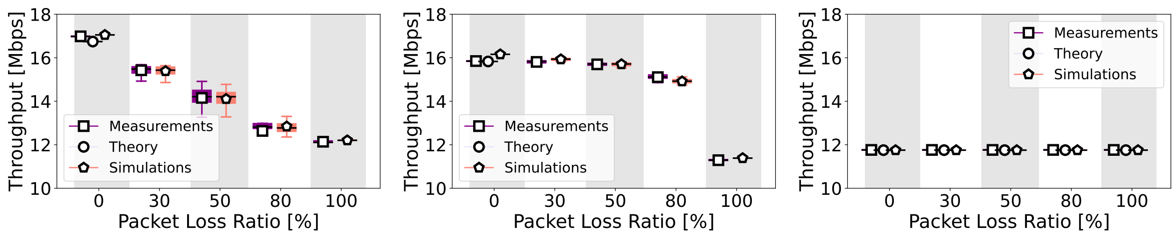
(a) 0% control packet losses. (b) 30% control packet losses. (c) 100% control packet losses.

Figure 5.8: Measured system throughput for 2 BSs over time considering various control packet losses, using a maxCQI-maxCQI policy.



(a) maxCQI-maxCQI case study. (b) maxCQI-maxCQI real traces. (c) RR-RR scheduling policy.

Figure 5.9: Throughput results for 2 BSs comparing theory, simulation outcomes and measurements for maxCQI-maxCQI and RR-RR scheduling policies for the case study and real UE traces.



(a) maxCQI-maxCQI case study. (b) maxCQI-maxCQI real traces. (c) RR-RR scheduling policy.

Figure 5.10: Throughput results for 3 BSs comparing theory, simulation outcomes and measurements for maxCQI-maxCQI and RR-RR scheduling policies for the case study and real UE traces.

RAN controller to the BSs. The goal is then to identify what is the influence of these control packet losses on the data plane, specifically the impact on the throughput.

5.2.7.3 Investigation of Sum Throughput Drop

First, we investigate the degradation in the sum throughput over all UEs for the example of 2 BSs with respect to the control packet loss ratio under the maxCQI-maxCQI scheduling policy. These results are shown in Fig. 5.8.

The results for 0% control packet losses, shown in Fig. 5.8a, reveal the ideal sum throughput given the CQI patterns, since there is no throughput deterioration. Alternatively, Fig. 5.8b shows the results pertaining to the case where the number of lost packets recorded is 30%. Hence, we can observe a slight throughput decrease between 50 s and 60 s from 18.3 Mbps to 11.3 Mbps. A further decrease is also observed between 200 s and 210 s to approximately 5.3 Mbps. The rationale behind this drop lies in the fact that when a control packet loss occurs, the previous resource allocation policy is applied from each BS. In that case, the BS whose UE has the lowest CQI gets all the resources and consequently, a lower throughput compared to the ideal case is observed.

The situation degrades even further when the packet losses increase to 100%. For a loss ratio of 100% only the first control decision is received correctly from BSs and all the others are considered lost, as demonstrated in Fig. 5.8c. In such a scenario, the instances where the throughput drops increase drastically. The worst experience is recorded from 230 s until the end of the measurements, where there is a drop to 11.3 Mbps compared to the maximum sum throughput of 18.3 Mbps, which represents an overall decrease of 38%.

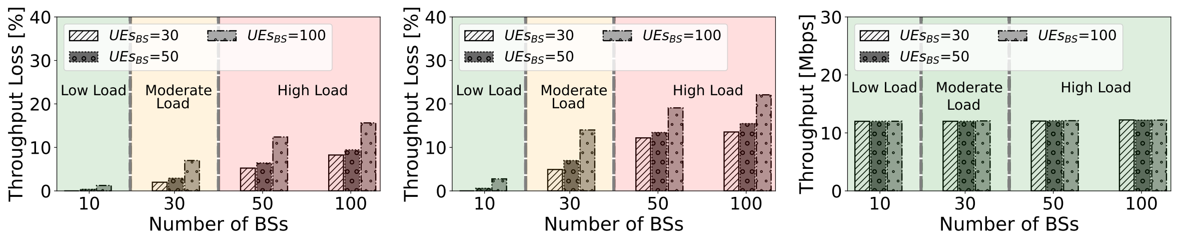
5.2.7.4 Throughput Evaluation: 2 BSs

So far, we have demonstrated results of the throughput degradation with respect to the % of control plane packet losses. Next, we show results for all the considered patterns not only obtained via measurements, but also analytically in line with the outcomes of Section 5.2.6, and simulations (as explained in Section 5.2.5). For all the cases we consider both maxCQI-maxCQI and RR-RR scheduling policies. The results for 2 BSs for the case study as well as for real UE traces (see Fig. 5.5) are shown in Fig. 5.9.

For the results of the maxCQI-maxCQI scheduling policy, depicted in Fig. 5.9a for the CQI case study, the sum throughput for the 2 BSs decreases almost linearly with the increase in control packet loss ratio. The simulation results, theoretical outcomes and measurements depict the same mean value. In that regard, we can conclude on the accurate prediction feature of our approach to capture correctly the effect on control packet losses in the data plane.

The best-case throughput in Fig. 5.9a is recorded, as expected, when there are no losses in the control plane, and its value is around 14.8 Mbps. On the other hand, when all the control packets are lost, the throughput drops below 12 Mbps. This leads to almost 20% in throughput degradation.

Similarly, Fig. 5.9b shows the throughput achieved for 2 BSs for maxCQI-maxCQI with real UE CQI traces. The maximum throughput is recorded for 0% losses with around 15 Mbps,



(a) maxCQI-maxCQI: 2 BSs. (b) maxCQI-maxCQI: 3 BSs. (c) RR-RR scheduling policy.

Figure 5.11: Throughput measurements for 2 BSs and 3 BSs while varying the number of emulated BSs and UEs for maxCQI-maxCQI and RR-RR scheduling policies.

whereas in the worst case its value is around 13.8 Mbps for 100% losses. Again, we stress that the results obtained from the measurements match the theoretical results and simulation outcomes, further demonstrating the correctness of our methodology when using real UE CQI traces as well.

Considering the RR-RR scheduling policy, as observed in Fig. 5.9c, the packet loss ratio does not have an impact on the overall system throughput. This is explained by the fact that BSs always receive equal amount of resources and thus when they do not get updates from the SD-RAN controller, they resume using the previous control decision. Since that decision is nonetheless the same by design, no throughput degradation occurs. As for both the case study and real traces the RR scheduler records identical results, we show the results only once. Compared to the maxCQI-maxCQI approach, the achieved throughput is 19% lower, i.e., $\sim$ 12 Mbps. Consequently, depending on the network conditions, a trade-off between robustness and optimality should be considered.

5.2.7.5 Throughput Evaluations: 3 BSs

Here, we portray results for 3 BSs for the case study as well as for real UE traces for maxCQI-maxCQI and RR-RR scheduling policies.

Again, the results of the CQI case study (Fig. 5.10a) and for real UE CQI traces (Fig. 5.10b) for the maxCQI-maxCQI scheduling policy exhibit an almost linear drop in the throughput with the packet loss ratio increase, similar to the 2 BS setup. Compared to the case with 2 BSs shown in Fig. 5.9a, the overall achieved throughput is higher for any configuration. The rationale behind this result lies in the higher number of BSs, which increases the chance that at least one of the BSs will experience higher throughput. In that case, even if a packet is lost, the overall system throughput does not decrease considerably. In both scenarios, the theoretical results are presented only for a packet loss ratio 0. Nonetheless, simulations are portrayed for all configurations. In any case, theory, simulations and measurements demonstrate almost equal averages. In terms of throughput degradation, in both scenarios for the case when all the control packets are lost, the throughput drops below 12 Mbps compared to 16 Mbps recorded with no losses. That leads to a throughput degradation of $\sim 29\%$.

For the RR-RR scheduling policy depicted in Fig. 5.10c, the throughput is robust to control packet losses and remains constant for all configurations and among BS setups to approximately 12 Mbps, demonstrating again the insensitivity of this scheduling policy to control packet losses.

5.2.7.6 SD-RAN Emulated Traffic

While previously we assumed an arbitrary packet loss ratio in the control plane to ease the theoretical evaluation, in this part, we drop this assumption in order to assess the evaluation of the system. To that end, we generate background traffic in the network with respect to BSs and UEs, using the source-code of `MARC` [Pap+21a]. For the remainder of this section, results are demonstrated against multiple BSs and UEs, but we only measure the throughput for 2 and 3 BSs OpenAirInterface containing up to 3 UEs each. The additional BSs and UEs generated by the SD-RAN traffic emulator are considered only as background traffic. E.g., if 50 BSs and 30 UEs in each BS are shown, 47 or 48 of those BSs are generated with the SD-RAN traffic emulator, the additional 2 or 3 ones (depending on the scenario) are generated with OpenAirInterface.

The results for 2 and 3 BSs are presented for real UE traces and are evaluated with respect to background traffic generated by emulated BSs and UEs. The outcomes are depicted in Fig. 5.11a and Fig. 5.11b for the throughput loss in % for the maxCQI-maxCQI scheduling policy for 2 and 3 BSs. In turn, Fig. 5.11c shows the achieved throughput for the RR-RR scheduling policy.

For both Fig. 5.11a and Fig.5.11b, the throughput loss in % increases as the number of BSs and UEs in the network increases. The figure has been split in three areas, namely low load (less than 1000 data plane elements, such as BSs and UEs), moderate load (maximum 3000 data plane elements), and high load with more than 3000 data plane elements. In the regime of low load, in both cases the throughput loss does not exceed 2%, indicating a relatively good operational region. Alternatively, in the moderate load regime, the degradation of the throughout increases up to 7% for 2 BSs and 13% for 3 BSs. This is due to the fact that more losses can occur as the number of BSs increases. Finally, the losses increase beyond 15% and reach almost 23% in the high load scenario for 2 and 3 BSs, accordingly. This indicates that the operation in this region is not satisfying for the network.

On the other hand, as already mentioned, for the RR scheduling policy, the throughput remains robust irrespective of the network load, as shown in Fig. 5.11c.

5.3 Effects of SD-RAN Control Plane Design on User QoS

In the previous section, an analytical method and measurements were portrayed for the effect of SD-RAN controller inconsistencies with respect to the UE throughput in the data plane. Yet, the results are demonstrated considering a single controller scenario. Moreover, the analysis is provided for traditional scheduling algorithms such as maxCQI and RR. In order to investigate further the effect of SD-RAN control plane design on the UE side, in this section, we extend our analysis towards a distributed SD-RAN control plane too. Further, we portray simulations for various QoS metrics apart from throughput, such as latency and packet drop rate, while considering more sophisticated scheduling algorithms as explained in the remainder of this section.

5.3.1 SD-RAN Concept

For our approach, we consider the SD-RAN concept based on ORAN [O-Ra], as illustrated in Fig. 5.2. Given that our main goal is to provide details with respect to the UE QoS, we investigate a near-RT RIC according to the ORAN terminology, which deals with functionalities critical to QoS such as resource scheduling. A near-RT RIC is represented with the `FlexRAN` [Fou+16] controller in the state of the art. In the `FlexRAN` platform, the E2 interface corresponds to the `FlexRAN` **protocol**, whereas eNBs/gNBs are equipped with a software referred to as the `FlexRAN` **agent**. In a single controller approach, only one `FlexRAN` controller is responsible for all the eNBs/gNBs in the network. The controller and agents maintain a TCP connection for statistics collection and control decisions enforcement. These rules can range from scheduling, UE handover to power management. For more information on the `FlexRAN` protocol, the reader is referred to Section 4.1.2.2.

Upon the initialization of the eNBs/gNBs, a database is created at the `FlexRAN` controller. This is known as RIB. The controller collects the periodical statistical reports sent from the agents regarding the state of attached UEs and stores this information in the RIB. These reports include UE CQIs, internal UE packet buffer status as well as wireless resource utilization. Notably, while increasing the number of eNBs/gNBs and UEs that a controller has to manage, the controller load increases, which in turn may lead to undesired system behavior. Thus, the introduction of a distributed control plane becomes necessary.

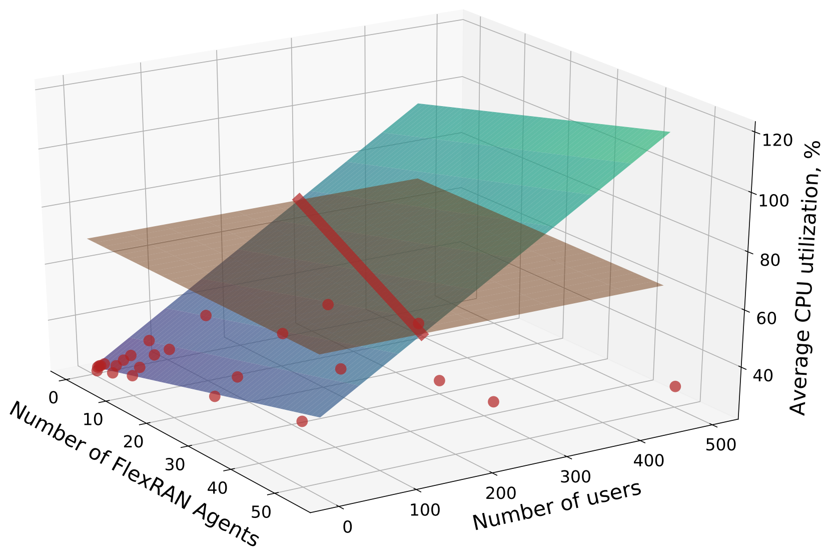


Figure 5.12: CPU utilization approximation based on the number of `FlexRAN` agents and UEs in the data plane, according to results obtained in [Pap+21a].

5.3.2 SD-RAN Control Plane Design

To overcome the scalability issue of the physically centralized control plane in SD-RAN, following [AWL15; Rig+14], we propose a physically distributed control plane SD-RAN. In addition, we exploit dynamic gNB-to-controller mapping to allow even re-distribution of the load among controllers. We again stress, that in this thesis the terms eNBs/gNBs and BSs as well as agents are interchangeable.

5.3.3 Control Handover

The assignment of BSs to `FlexRAN` controllers is realized based on a load-balancing application similar to Elasticon [Dix+]. The algorithm is periodically executed to evenly distribute the network load among SD-RAN controllers, depending on the current UE traffic and wireless conditions. When the BS is assigned to a new controller, a **BS control handover** occurs. We note that from now on, a BS control handover refers to the handover of a single BS, whereas control handover refers to the overall network handover procedure. The target controller requests the database of the migrating BS from the initial controller. After the database transmission, the target controller contains the statistics of the BS and it is now in charge of the management.

The **BS control handover** involves a) the reading of the eNB/gNB-migrated database from the RIB of the initial controller, b) sending of the database to the target controller, and c) writing of the database to the RIB of the target controller. These processes impose an additional load

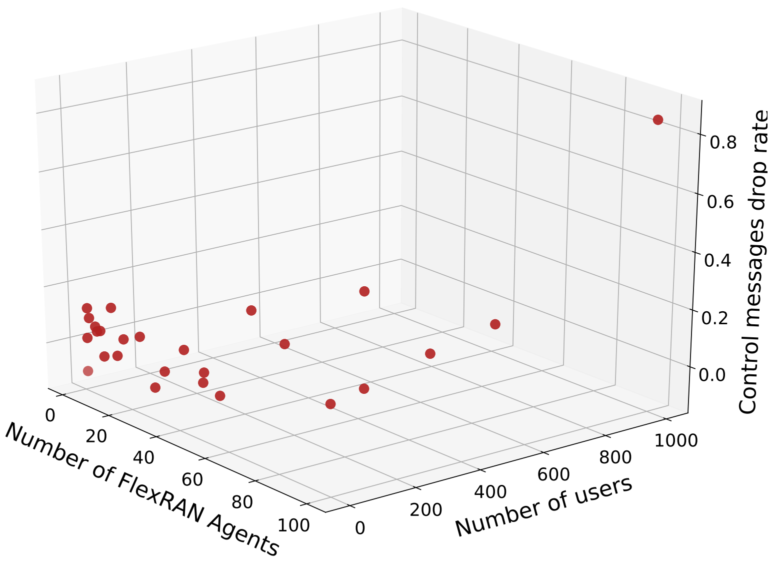


Figure 5.13: Control messages drop rate as a function of the number of `FlexRAN` agents and UEs in the data plane.

on the controller hardware. We express the RIB stored data (in KB) as

$$RIB_{load} = \sum_{i=1}^{m}(2 + 3n_{UEs,i}), \tag{5.24}$$

where m and $n_{UEs,i}$ are the number of agents and UEs of agent i, respectively. Part of the RIB corresponding to the migrating BSs is transferred during a control handover.

5.3.4 Modeling Controller Load from Measurements

In order to model the controller overhead during a handover, we take over measurements from our SD-RAN controller benchmark, `MARC`, presented in Chapter 4 as a basis. Fig. 5.12 gives the CPU utilization of the `FlexRAN` controller as a function of the number of attached `FlexRAN` agents and UEs. The orange dots represent the experimental points from the results presented in Chapter 4, whereas the blue-green plane shows the best-fit approximation performed on the experimental dots. As illustrated in Fig. 5.12, the CPU utilization of a single controller increases with the number of connected devices, until it reaches a maximum of $\sim 70\%$ (highlighted with the orange horizontal plane), after which experimental points show a decrease, below 40%, corroborating the controller undesired behavior, which leads to control packets being lost.

The outliers in Fig. 5.12 pertain to the case when the controller is overloaded. Nevertheless, our model considers only the region of controller utilization that goes up to 100%.

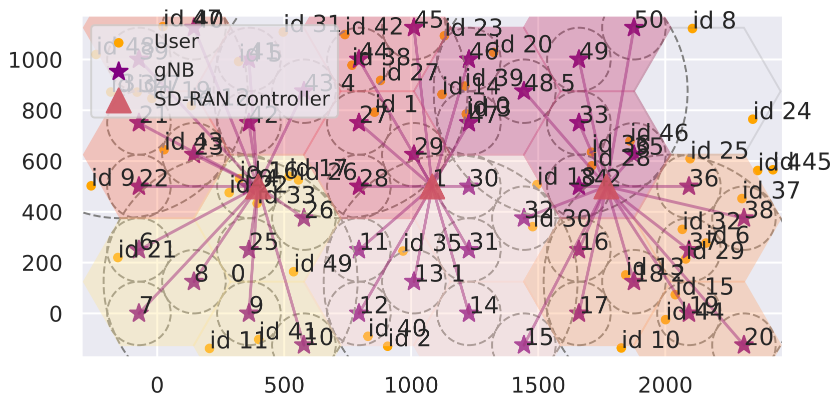


Figure 5.14: SD-RAN simulator topology consisting of SD-RAN controllers, gNBs and UEs.

Analyzing the difference between the expected and measured control messages reception rate of SD-RAN controllers reported in Section 4.5.5 and the best-fit approximation shown in Fig. 5.12, we obtain the model of the control messages drop rate. This is shown in Fig. 5.13 as a function of the number of `FlexRAN` agents and UEs. For our operation region of interest in this section (up to 300 UEs and 16 BSs), the drop rate remains below 20%.

The BSs' databases' transmission, taking place during the control handover, effectively increase the CPU utilization of the initial and the target controllers. With the increase in CPU utilization, the control messages drop rate also increases. Eventually, the control handover results in a higher drop rate of control messages, including statistical messages, and stagnates the control efficiency. Using measurements with `FlexRAN` we can neglect the impact of TCP losses between the controller and agents, because the amount of data that is transmitted for a single agent does not exceed 2.5 Mbps, even if the number of UEs is higher than 100 per agent (see Eq. (5.24)). That data rate is much lower than the usual dedicated link capacity between an agent and the controller in our testbed ($\sim$ 100 Mbps). Thus, only the CPU is the source of performance deterioration. Additional insights are provided in Section 5.3.6 of this chapter.

5.3.5 SD-RAN Simulator

In this section, we first present an overview of the simulator structure. This is proceeded by the description of the SD-RAN scheduling architecture and scheduling policies. Finally, we elaborate on the control handover procedure.

5.3.5.1 Simulator Structure

The influence of the control handover on the UE QoS is studied with the help of the SD-RAN simulator. The details and overview of the simulator are provided as below.

The simulator is developed in Python and follows a time-based approach with a minimum granularity corresponding to a TTI of 1 ms. There is a clear separation of the control plane, where the SD-RAN functions reside, and data plane, where the gNB and UE-related functions are placed.

The device class represents UEs and is implemented in accordance with 3GPP specifications [3GP18a; 3GP17d]. The packet inter-arrival times on devices follow an exponential distribution, whereas UEs move according to the Random Waypoint mobility model [RS11]. Similarly, the implementation of the gNB class follows the specifications [3GP17a; 3GP17b]. UE handovers, managed by gNBs are taken from [3GP18b], whereas resource scheduling is performed across time and frequency, with PRBs as the unit of resource allocation. All wireless channel characteristics are based on [3GP17c].

The simulations represent an outdoor scenario with gNBs generated according to [3GP19a] as portrayed in Fig. 5.14. SD-RAN controllers are located in the center of the topology. Every SD-RAN controller is in charge of a set of gNBs depending on the load balancing algorithm similar to [Dix+]. All gNBs under the SD-RAN controller's management share resources, yet gNBs controlled by different controllers do not interfere.

5.3.5.2 SD-RAN Scheduling Architecture

In our approach, we envision the RAN scheduling to be performed in a hierarchical manner similar to [CNS18; Pap+19b]. The SD-RAN controller distributes wireless resources which we refer to as PRBs to the underlying gNBs in what called **control scheduling**. For the remainder of this chapter, the words resources and PRBs are interchangeable. The **control scheduling** is based on channel statistics reported to the SD-RAN controller by gNBs in form of frequent periodical statistical reports every 1 ms. These messages contain information with respect to current internal packet buffer sizes and the CQI of every connected UE. We refer to these messages as **control messages**. The **control scheduling** happens every 10 ms, known as **control scheduling period**. Within this period, the PRBs assigned to gNBs remain static.

Based on the resource distribution from the SD-RAN controller among gNBs, each gNB allocates the received resources to its respective UEs every 1 ms, which we call **MAC scheduling**. The rationale behind **control scheduling** frequency being larger than **MAC scheduling** frequency lies on the communication overhead among controllers and gNBs. Due to the fact that **MAC scheduling** needs to be performed every 1 ms or less in 5G, such time criticality does not allow for frequent **control scheduling** updates as it will influence the timely operation of **MAC scheduling**. Note that every statistical message and the **control scheduling** decision

Algorithm 3 Enhanced Scheduling

```
if $TTI \ \% \ scheduling\_periodicity == 0$ then
    for $cntrl$ in $controllers$ do
        $p$ = RANDOM()
        if $p \leq 1 - P_{loss}(UEs, agents)[cntrl]$ then
            $allocation[cntrl] \leftarrow NULL$
            $buf\_list\_tmp[cntrl] \leftarrow buf\_list[cntrl]$
            for $j$ in range $available\_resources$ do
                $ue\_sequence \leftarrow NULL$
                for $k$ in range $scheduling\_periodicity$ do
                    $best\_ue \leftarrow$FIND_BEST_UE($k$,
                    $buf\_list\_tmp[cntrl]$,
                    $sinr\_list[cntrl][TTI + k]$)
                    $buf\_list\_tmp[cntrl] \leftarrow$ UPD_BUF()
                    $ue\_sequence$.INSERT($best\_ue$)
                end for
                $allocation[cntrl]$.INSERT($ue\_sequence$)
            end for
        end if
    end for
end if
```

itself can be lost with probability $P_{loss}(UEs, agents)$, which is shown in Fig. 5.13 as control message drop rate.

5.3.5.3 Control Scheduling Policies

In our simulations, we propose and analyze various control scheduling policies to demonstrate the effect on the resource allocation optimality.

The initial **control scheduling** policy is based on the maxCQI principle. We utilize the maxCQI approach both for the **control scheduling** that runs on the SD-RAN controller, as well as for the **MAC scheduling** on gNBs.

When the SD-RAN controller only bases its decision on the UE reports obtained in the last ms, we call this scheduling policy **current-based maxCQI**. While this policy may require less storage in the controller's RIB since it only stores the last UE statistics, it does not capture all channel variations. For instance, if more resources are required or fewer resources are needed until the next **control scheduling** period, then some UEs are not able to receive the adequate amount of resources they require in the future.

Alternatively, the SD-RAN controller may store all the UE statistics reports in the last 10 ms. We refer to this policy as **past-based maxCQI**. While this requires more storage on the controller side, it leads to a more optimal solution. For instance, an average buffer and CQI information based on the last 10 ms can capture a better view of the channel characteristics and UEs' requirements.

However, this solution is still not optimal, as it does not account for the future buffer and CQI dynamics. A further step towards optimality assumes a prediction of average UE buffer size and CQI values for the remaining 10 ms until the next **control scheduling** takes place. Thus, more resources would be given to those BSs, whose UEs will require them in the future. We refer to this policy as **future-based maxCQI**.

The second type of scheduler is the **enhanced scheduling**. It shares similarities to the **future-based maxCQI**, however, it contains a full picture of the buffer dynamics and CQI values of each UE for the whole control scheduling period (i.e., 10 ms) instead of averages. Having the exact predicted statistic dynamics for the whole next scheduling period, optimal decisions can be established. In reality, such details are not available to the SD-RAN controller, however, utilizing machine learning for prediction, this can be provided [AGK19].

For every available PRB, Alg. 3 finds the optimal sequence $ue\_sequence$, which indicates to which UE the given PRB should be assigned at every TTI within the scheduling period. The best UE is determined with the function FIND_BEST_UE(). It assigns the resource to every UE, and knowing the exact buffer statistics and future CQI at this time instance, it calculates how many packets can be processed with such a configuration. The UE which provides the maximum number is chosen. With the resource being assigned to a device, the effective prediction of buffer size is changed. This is captured by the UPD_BUF() function. More precisely, some of the buffer sizes reduce due to packets being sent with the already assigned PRBs, and these packets should not be served again. For every controller, the complexity of the algorithm in every TTI is directly proportional to the product of the number of PRBs and number of UEs managed by that controller.

5.3.5.4 Control Handover

The considered architecture envisions a distributed control plane with dynamic gNB-to-controller mapping, aiming at balancing controllers and preventing them from being overloaded.

The Allocation class of the simulator implements the load balancing algorithm and its functionalities. We adopt the load balancing algorithm from [Dix+], which equalizes the CPU utilization of controllers. The Allocation application periodically triggers the controllers to send the information about gNBs and UEs attached to them. After gathering all the statistics, the load balancing is performed. This results in an optimal mapping of gNBs to controllers, reported to all the controllers, which now can enforce the new decision. Thus, the control over a gNB should be handed over to the new target controller. Regarding the control plane consistency during handovers, we rely on solutions applied in core networks [SK17]. In this thesis, we do not consider it as it does not bring any technical novelty. Instead, those models from the literature could be incorporated in RAN, as future work, to maintain the control plane consistency.

To start the handover procedure, the target controller should request the database of the migrating gNB from the initial controller, containing the control scheduling relevant information, such as UE buffer state and CQI statistics.

During the database transmission, the gNB keeps being managed by the initial controller, with the control handed over to the target controller when the database transfer is finished. To reflect the overheads caused by the control handover, both controllers receiving and transmitting the database of the migrating gNB are modeled to have an increase in CPU utilization. In our simulations, we utilize values for the CPU increase and BS database transmission based on real measurements performed with `FlexRAN` [Fou+16]. While the CPU increases as an effect of the increase of the number of UEs and agents, the control packet drop rate increases. To avoid a controller from transmitting/receiving several databases simultaneously, database transmission is scheduled sequentially.

5.3.6 Performance evaluation

This section represents the numerical results obtained with the 5G-based SD-RAN simulator described above. Unless stated otherwise, the system bandwidth corresponds to 20 MHz per SD-RAN controller shared among the managed gNBs. This corresponds to a total amount of 50 PRBs of 30 KHz sub-carrier spacing every 0.5 ms, whereas the central frequency of gNBs is 4 GHz. The control scheduling is performed every 10 ms, whereas the MAC scheduling takes place every 1 ms as elaborated in Section 5.3.5.2. The arrival rate of packets per UE corresponds to 0.4 ms^{-1}, of size 600 bytes. All simulation parameters are based on [3GP17a]. In the remainder of this subsection, we provide results for the single controller as well as a comparison with the distributed approach with respect to average UE packet delay, throughput, and packet loss. Finally, we demonstrate results concerning handover effects in a multi-controller scenario.

In general, different traffic models affect differently the performance of the controller and thus, the data plane. For instance, considering a low UE mobility and low UE density scenario, means that the SD-RAN controller will not be overloaded and therefore fewer handovers will

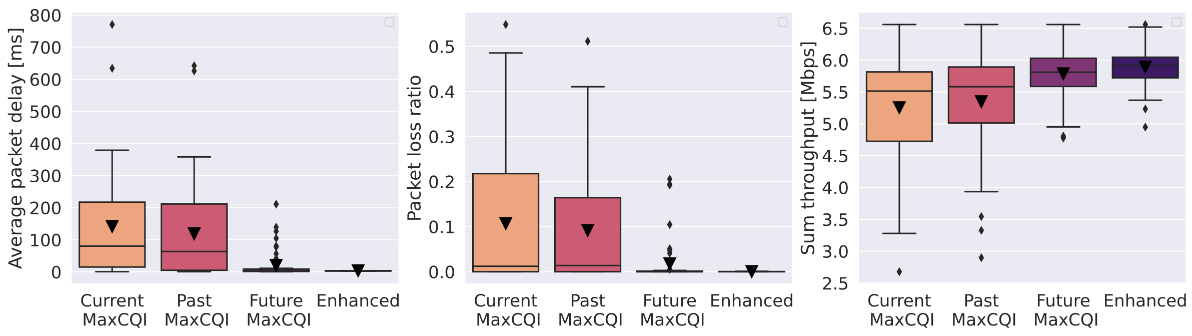


Figure 5.15: Comparison of various control scheduling policies with respect to the average packet delay per UE, packet loss ratio and sum throughput for a single controller use case with no control messages drops.

occur. This is not of any practical concern. Consequently, it is not shown in our results. On the other hand, a high UE density and mobility means that the SD-RAN controller is stressed to its operation limits. This factor combined with the scarcity of the wireless resources presents irregularities in the data plane operation. That is the reason why we focus on the high-density scenario in this chapter.

5.3.7 Single Controller Setup

Initial experiments consider the control plane containing a single controller. There are in total 8 gNBs and 200 UEs randomly distributed into an area of 1900 × 800 m. The SD-RAN controller is located in the center of the topology. The simulation duration is 2000 TTIs (i.e., 200 control scheduling periods), sufficient to demystify the impact on the data plane.

5.3.7.1 Scheduling Policies Comparison

The considered schedulers are: 1) maxCQI Current-based scheduler, 2) maxCQI Past-based scheduler, 3) maxCQI Future-based scheduler, 4) Enhanced scheduler.

Initially, the proposed control schedulers are compared for the single controller scenario with the assumption that no control messages are lost to solely capture the effect of control scheduling principles. These results are portrayed in Fig. 5.15. The left and middle boxplots represent the average packet delay per UE, expressed in ms and the average packet loss ratio, whereas the plot on the right the UE sum throughput in Mbps. The packet delay is obtained between the timestamp of each packet arrival and the moment when the packet leaves the internal buffer of the corresponding UE. Additionally, for the packet loss ratio, the packets are considered as lost in two cases: when the maximum buffer size of 480 KB [3GP17a] is reached and when packets are left in the device buffer at the end of the simulation due to lack of resources.

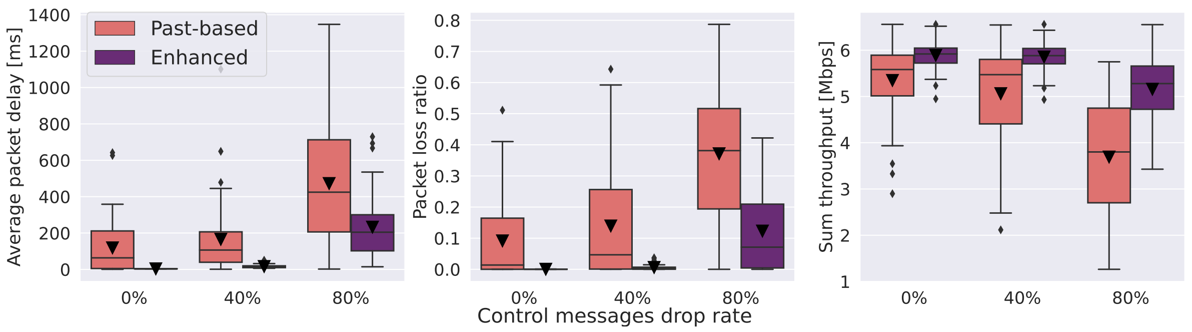


Figure 5.16: Comparison of various control scheduling policies with respect to the average packet delay per UE, packet loss ratio and sum throughput for a single controller use case with varying control message drop rate.

From Fig. 5.15, we can conclude that taking average past buffer values rather than the instant ones improves the scheduling in terms of delay and packet loss ratio, mainly due to less wasted resources within the scheduling period. The spared PRBs in **past-based maxCQI** can be assigned to UEs having a worse channel, decreasing the average delay and allowing more packets to be served. Utilizing the average future values is even more beneficial as it prioritizes UEs which are expecting more packet arrivals during the next scheduling period. As a result the **future-based maxCQI** scheduler shows improvements compared to the **current** and **past-based** schedulers on all considered metrics as illustrated in Fig. 5.15. In turn, the **enhanced scheduler** demonstrates equal average packet delay, while surpassing the best performing alternative (i.e., **future-based maxCQI**) by 12× in terms of packet loss ratio and by 2% in terms of throughput. This occurs as it assigns resources to UEs optimally, giving the exact number of PRBs required to send the buffered packets for the current channel conditions.

For the remainder of the section, only the results for past-based maxCQI and enhanced schedulers are considered. Whereas the enhanced scheduler obtains the best results overall, it is based on assumptions that include advanced prediction techniques. Alternatively, the past-based maxCQI scheduler is chosen as the benchmark that represents a common policy in current networks.

5.3.7.2 Analysis of Single Controller Performance

The number of UEs and gNBs for a single controller use case is fixed. Thus, to capture the effects of single controller undesired behavior on UE QoS due to lost control messages, we assume a varying control message drop rate. This variable can be fine-tuned at the start of the simulation and remains constant throughout it. The results for this setup are represented in Fig. 5.16.

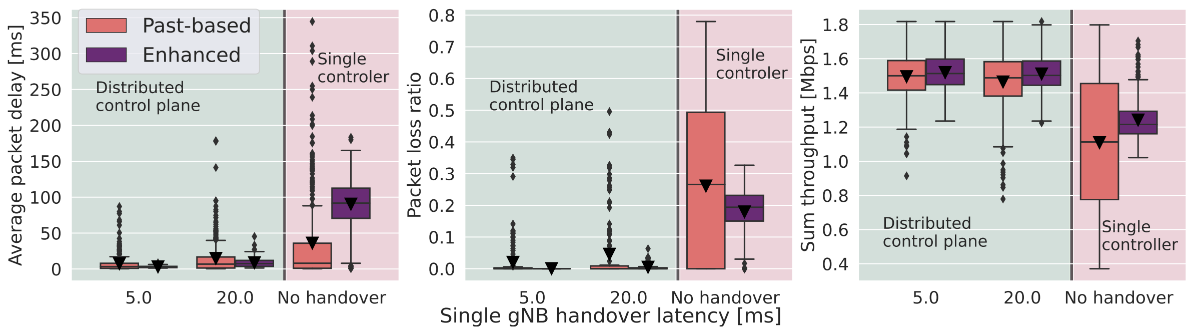


Figure 5.17: Comparison of the single controller use case referred to as no handover and distributed control plane scenarios for two gNB control handover latencies, representing a gNB with low number of UEs (5 ms) and a gNB with large number of UEs (20 ms). Results are illustrated for average packet delay per UE, packet loss ratio and sum throughput.

The left, middle and right plots portray how the packet delay, packet loss ratio and throughput change with increasing control messages drop rate. Overall, all aforementioned metrics increase with increasing control message drop rate, indicating that control scheduling optimality is compromised when UE statistics are not fully available at the controller. For the worst-case scenario of a control message drop rate (80%), the enhanced scheduler records on average 230 ms of packet delay compared to 470 ms achieved with the past-based scheduler. Additionally, the enhanced scheduler achieves 12% of packet loss ratio compared to 40% of the past-based. Finally, the UE throughput of the enhanced scheduler (5.2 Mbps) outperforms the past-based one (3.7 Mbps). The rationale behind this is twofold. Firstly, the UEs which require more resources due to their large buffers do not receive adequate amounts. Secondly, several UEs receive a larger amount of PRBs even when not needed due to outdated statistics. Hence, UEs with worse channels may not get enough resources.

5.3.8 Comparison of Single and Distributed Scenarios

The next experiments target the analysis of the impact of control handover on UE QoS. In this setup, there are 300 UEs, 18 gNBs and three SD-RAN controllers as shown in Fig. 5.14. As aforementioned each SD-RAN controller possesses 20 MHz of bandwidth. Additionally, the control messages drop rate, $P_{loss}(UEs, agents)$, is now deduced from the actual CPU utilization of the controller, which depends on the number of attached devices and the handover status taken from Fig. 5.12 and Fig. 5.13, respectively, and elaborated in Section 5.3.2.

The control handover routine is triggered every 400 ms. The communication between the SD-RAN controller and the load balancing application is TCP-based, with 80 ms delay in each direction, whereas the optimization based on Elasticon's [Dix+] load balancing algorithm [Dix+] lasts for 100 ms. For a single gNB control handover latency, we consider two cases a) 5 ms, representing a gNB with a database consisting of a low number of UEs (i.e., $\sim$ 10) and b) 20 ms, representing a gNB with a database consisting of a high number of UEs (i.e., $\sim$ 40). During the gNB control handover, the CPU utilization of both initial and target controllers is additionally increased by 30%. We stress that all the aforementioned values are based on measurements with the `FlexRAN` controller [Fou+16].

5.3.8.1 Effect of Handover on UE QoS

At the beginning of the simulation, the optimization algorithm is performed, resulting in an optimal allocation of gNBs to SD-RAN controllers. However, during the simulation, UEs move, attaching to gNBs controlled by different controllers than the initial one. As a result, even though the gNBs do not attach to new controllers, the load of the SD-RAN controllers dynamically changes because some gNBs become busier than the others. If the optimization algorithm is not applied, no control handovers occur, resulting in the case where a single SD-RAN controller takes over the whole control of the network and the other two controllers are idle. This scenario represents the single controller use case and is referred to as **no handover** in our results. Alternatively, a load balancing algorithm is applied that results in control handovers. The goal is to analyze the impact of the aforementioned approaches on the UE QoS.

The left, middle, and the right plots in Fig. 5.17 represent the average packet delay, packet loss ratio, and sum throughput per UE for two control handover latencies and for the system with the single SD-RAN controller referred to as **no handover**. From the obtained results, the average packet delay, packet loss ratio and throughput increase with increasing gNB control handover latency due to the increased period, during which the CPU utilization and control messages drop rate increase.

While the negative effect of gNB control handover latency is evident in the results presented in Fig. 5.17, yet the UEs in the case where no control handover is performed experience worse QoS. Compared to the worst-case gNB control handover latency (i.e., 20 ms), representing the distributed control plane use case, UEs in the single controller scenario (i.e., no handover) experience at least 20% lower throughput, 140% higher delay and 5$\times$ more packet drops on average. The reason for this behavior lies is in UEs' mobility, which cause a larger CPU utilization for one controller, while leaving others unoccupied. In this case, UEs not only have to share less amount of resources, but also increase the data the controller processes, resulting in higher dropped control messages.

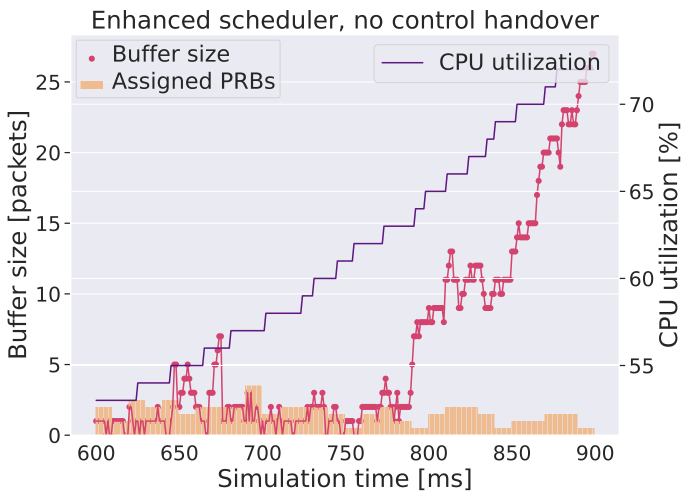


Figure 5.18: Time-series of buffer size, assigned PRBs and CPU utilization of the managing controller for one UE under a single controller use case. The total number of UEs for a controller increases over time from 100 to 300 UEs.

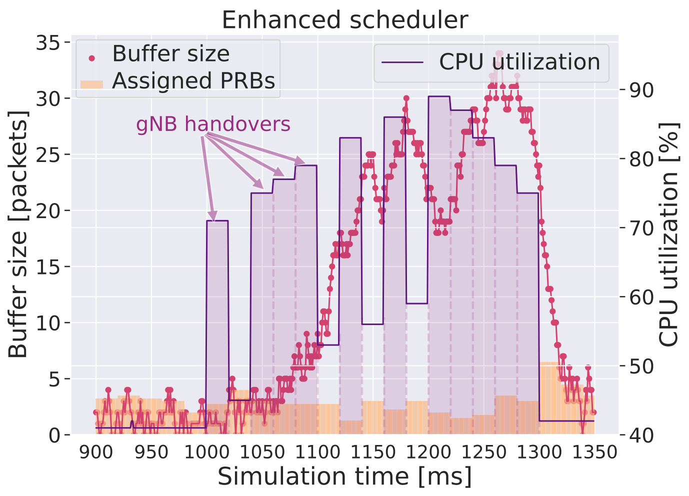


Figure 5.19: Time-series of buffer size, assigned PRBs and CPU utilization of the managing controller for one UE under a distributed control plane scenario with constant gNB control handovers.

5.3.8.2 Analysis of CPU Effect on SD-RAN Control Plane

As shown in Fig. 5.17, the distributed control plane outperforms the single controller use case with respect to UE packet delay, packet loss ratio and sum throughput. In this subsection, we provide insights on the rationale behind this performance degradation related to the CPU utilization of controllers.

The described effect can be observed in Fig. 5.18, which represents the time series of the buffer size of one of the UEs with the pink line, the number of PRBs scheduled to this UE with orange bars, and the CPU utilization of the implicitly managing controller with the purple line over the simulation time for a single controller scenario. It can be observed that the CPU utilization of the controller is constantly increasing due to more and more UEs assigned to it. Starting from TTI 600 with 100 UEs, the number increases to 300 at TTI 900. This results in a) fewer PRBs available to the UE, because the resources are shared between many UEs, and b) a higher number of dropped control messages due to an increase in the number of UEs and agents which imposes higher CPU load. Indeed, as observed in Fig. 5.18, the CPU load of the controller increases with time, which results in larger UE buffer size and less PRBs assigned.

In contrast, Fig. 5.19 demonstrates the time series of the current buffer size of one UE for a distributed control plane scenario. In this case, the CPU utilization fluctuates from TTI 1000 until TTI 1300 demonstrating control handovers. The CPU increases in chunks of 20 ms (i.e., worst-case gNB control handover latency). It starts from ~ 40% and increases up to ~ 70% from TTI 1000 to 1020. While consecutive handovers occur, the CPU reaches up to ~ 90% at TTI 1200 and then drops to ~ 40% when all the migrating gNBs have performed their handover. Although the CPU increases similarly to the single controller scenario, in this case, the buffer length of the UE follows the CPU pattern. Furthermore, the adequate amount of PRBs needed by the UE is re-established once the control handover terminates.

5.4 Summary

In this chapter, we investigated the impact of the imperfections of the SD-RAN control plane on the data plane. More specifically, we demonstrated firstly the throughout degradation of UEs in the data plane as a function of a) assumed control packet losses, and b) SD-RAN emulated traffic on the controllers through `Delphi`. For the former, we provided mathematical models for 2 and 3 BSs to predict the throughput. We ran extensive simulations and measurements with input parameters from real-life traces to verify our analytical models. While considering two scheduling policies, namely maxCQI-maxCQI and RR-RR, we showed how much the control plane packet losses influence the throughput of individual UEs, and hence the system throughput.

By and large, the results from the proposed mathematical model match those from simulations and measurements, proving the validity of our model. Moreover, the insights we provide hint that when a maxCQI-maxCQI scheduler policy is used, a high load on the network control significantly deteriorates the throughput (by almost 20%). Contrary to that, RR-RR scheduler is robust to such effects. On the other hand, the maxCQI-maxCQI scheduler achieves higher overall throughput. Therefore, depending on the network conditions, the appropriate resource allocation scheme should be used.

`Delphi` provides the first step towards performance prediction in SD-RAN networks, while demystifying important aspects of the system with high load. Moreover, despite the fact that we provide an analytical method for the case when control packets are assumed lost, a similar approach can be used for the case when the packets are not considered lost, but delayed. For this, measurements need to be conducted to identify the average packet delay occurrence within the control period. In that case, the analytical methods presented in Section 5.2.6 can be utilized.

An alternative avenue of research concerns mechanisms that make the system more robust to changes while maintaining high throughput. A potential solution would be to introduce admission control mechanisms or distributed control plane for SD-RAN controllers.

Additionally, in this chapter, we evaluated the effect of the SD-RAN control plane design on the UE QoS performance for various metrics such as UE throughput, latency and packet drop rate through the use of a 5G-based SD-RAN simulator. All evaluations are portrayed for both a single controller and a distributed SD-RAN control plane. The design characteristics of the SD-RAN control plane are based on real measurements with `FlexRAN` and 3GPP standardization. In that regard, we demonstrated that a single SD-RAN controller cannot handle a large number of devices in the data plane as it degrades the performance of the UEs by 20% in terms of throughput, 140% in terms of average delay and 5× in terms of packet loss ratio, which can be overcome with a distributed control plane. However, the introduction of such a system induces complexity and overhead on the control plane. Yet, the distributed control plane with control handovers still outperforms the single controller design in terms of the UE QoS.

Chapter 6

System Design and Optimization of Multi-Network Slicing

In the previous chapters of this thesis, the main focus lied in the RAN side of the cellular network. However, if we consider the 5G network requirements in terms of ubiquity and extension of service coverage, investigation on system design concepts and performance evaluation of multi-network environments that involve also satellite and aeronautical networks, becomes necessary. This is considered in this chapter.

Ubiquity is foreseen as one of the main challenges of next generation networks, especially since now users demand to be connected everywhere, while still maintaining high data rates and low latencies. In that regard, the integration of satellite and terrestrial networks [Boe+18; Cal+20] is becoming crucial to accommodate such requirements for network environments like aircrafts or trains [GKP18; Var+19]. However, given the distinct nature of applications in 5G and the limited backhaul capacity offered by satellite links [Gog], current satellite-terrestrial network infrastructures urge for more dynamic techniques for the control and management of network resources.

An interesting 5G use case in a multi-network setup involves IFECS, where the market is expected to reach USD 7.65 billion by 2023 [Mar]. For IFECS, the complexity of resource management grows due to limited backhaul capacity (e.g., a satellite link [Gog] or a Direct Air-to-Ground (DA2G) link [Inm]) and large round trip delay, (e.g., the delay of a Geostationary Equatorial Orbit (GEO) satellite link is $\sim$ 200 ms). Additionally, the possibility to store large data volumes in the aircraft is restricted, leading to reduced cache sizes, while the traffic for capacity-demanding and heterogeneous applications on board (e.g., video streaming, Voice Over-IP (VoIP) calls and web browsing) is growing.

On the one hand, to cater for high round trip delays, the enhancement of the RAN with storing and computing capabilities that reduces the usage of the limited backhaul link is suggested [Wan+14]. On the other hand, 5G-related solutions that ease the management and orchestration of the limited resource bottleneck in aircrafts are emerging. Among these, the most well known are SDN and network slicing.

Previous works [Vo+18; WGT19; TL18] that considered network slicing for both cache and backhaul links assume that the user file request profile, which expresses the PMF of all files requested by a user, is shared with the slice manager (Section 2.3.2) as part of the slice request. This is an assumption that may not be feasible for some slices, especially those with privacy concerns. SPs may not have access or may not be willing to share such data. We will refer to these slice types as **uncooperative** slices. Moreover, some slices may not have access to a user profile but may share the anonymous user profile statistics over time. We call these slices **semi-cooperative**. Finally, we refer to the slices that are willing to share the real profile of users as **cooperative**. From the optimization point of view, maximizing the number of accommodated slices in the network cannot be achieved in the case of many uncooperative slices. Furthermore, the number of slices served is minimal if all slices are uncooperative.

With the previous facts in mind, it is of utmost importance to investigate the cost of a slice with respect to the slice type and incorporate that to the network slicing problem. In that regard, in this chapter, we aim at identifying the cost of a network slice initially and include it in our approach. Our goal is to maximize the overall number of served slices, while also allowing slices that are not willing to share user details with the SPs (i.e., uncooperative slices) to be served. The main contributions of this chapter are:

- Performance evaluation for the cost of a network slice with respect to cache and backhaul resources for IFECS, while distinguishing among cooperative, semi-cooperative and uncooperative slices with different slice profiles.
- The introduction of a dynamic approach that investigates the effect of time in the performance of the network slicing algorithms in case an adaptation is required in the statistics gained from user profiles within slices.
- The design and evaluation of a Mixed-Integer Nonlinear Programming (MINLP) whose objective is to maximize the number of served slices for IFECS and thus, increase the profit of SPs.

Content and outline of this chapter: Section 6.1 discusses related work. Section 6.2 introduces the system model and details the problem formulation. The main findings regarding the cost of a network slice with respect to backhaul and cache resources as well as results of the optimization problem are presented in Section 6.3. Section 6.4 highlights emerging 6G applications with respect to multi-network environments. Finally, a summary of this chapter is provided in Section 6.5. The content of this chapter is based on the results presented in the following publication:

[Pap+21b] A. Papa, H. M. Gürsu, L. Goratti, T. Rasheed, W. Kellerer. "Cost of Network Slice Collaboration: Edge Network Slicing for In-Flight Connectivity." In: Proc. of the IEEE International Conference on Communications (ICC). 2021.doi:10.1109/ICC42927.2021.9500820.

6.1 Related Work

The problem we tackle here mainly concerns network slicing considering multiple resources. In principle, multi-resource network slicing consists of RAN slicing, cache or computing slicing and backhaul slicing (i.e., satellite links). Previous works tackle RAN slicing [Pap+19b; Man+19], where an efficient allocation of the radio resources is achieved while maintaining isolation and QoS. Yet, none of the aforementioned works considers content storing or backhaul allocation for the network slicing problem. In our scenario, RAN slicing can be tackled by applying more sophisticated wireless technologies inside the aircraft such as optical fiber or Light Fidelity (LiFi). Although the efficient use of the scarce wireless resources is one of the most important challenges in a wireless scenario, studies in the literature demonstrate that effective caching techniques in the wireless access points can enhance the network performance and reduce the backhaul usage [Wan+14; BBD14]. Recent papers [Vo+18; WGT19; TL18; XCZ18; DOr+20] consider also cache storing and task offloading for the network slicing problem. Whereas the proposed solutions are relevant and provide interesting insights on the joint allocation problem, they do not consider a constrained backhaul capacity, such as a satellite link, and they assume perfect knowledge about the user profile of a slice.

Content popularity prediction is proven challenging especially because the user profile changes dynamically with time and it is not known beforehand [Yan+18]. In our work, we also do not assume that the user profile is known and an interaction with network slices is required. The interaction concerns building of such a profile dynamically and investigates the time required as a function of resource utilization until correct estimation of the user profile is obtained.

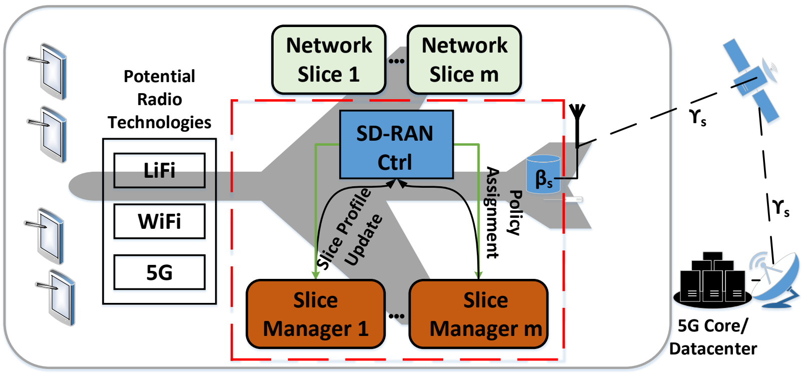


Figure 6.1: SD-RAN platform architecture. An SD-RAN controller is envisioned for the management of the cache and backhaul resource distribution. Each network slice consists of a slice manager that exchanges statistics with the SD-RAN controller for the resource allocation.

Motivated by the user behavior, we investigate the definition of the cost of a network slice with respect to the amount of cache and backhaul resources required to achieve the goal. A similar approach is followed in [KGD18] for identifying the cost of caching between the network and SPs. Additionally, we consider the cost of backhaul resources in our model since it is the critical bottleneck for aircraft communications.

6.2 System Model and Problem Formulation

We propose a model inside an aircraft where the components of the IFECS system, wireless connectivity, cache and backhaul link are called the infrastructure, and are managed by the InP through an SD-RAN controller, as shown in Fig. 6.1. Among the radio technologies, one can choose between Wireless Fidelity (WiFi), LiFi or 5G for the wireless connectivity. In the latter, the 5G RAN is deployed in the aircraft or satellites [GKP18], whereas the 5G core is deployed on the ground.

Moreover, the infrastructure contains an internal cache that can store up to B files and a backhaul link of capacity Γ (i.e., a satellite link). The infrastructure is leased to a set of network slices $\mathcal{S}$ of size m, with cache and backhaul resources assigned to them. These resources are denoted by γ_s and β_s, respectively, and specify percentages from the total capacity. In the described setup, the SD-RAN controller is in charge of deploying and assigning these resources to slices. For each accepted network slice, the SD-RAN controller creates a slice manager. The latter manages the users within a slice. It is responsible for accepting the users in the system, exchanging information with the SD-RAN controller about the users and distributing the received resources within the slice's domain. The information regarding the users and their file preference profiles is not always shared between the slice managers

Table 6.1: System parameters and variables

Variable	Description
$\mathcal{S}$	Set of network slices
$\mathcal{U}_s$	Set of users within a network slice
$\mathcal{K}$	Set of files in the system
$p_{u,f}$	Request probability of user $u \in \mathcal{U}_s$ for file $f \in \mathcal{K}$
z	Zipf distribution design parameter
B	Cache size
Γ	Backhaul capacity
L	File size
$\lambda_{u,f}$	Request rate of user $u \in \mathcal{U}_s$ for file $f \in \mathcal{K}$
$h_{s,f}$	Hit probability of slice $s \in \mathcal{S}$ for file $f \in \mathcal{K}$
λ_u^{hit}	Cache hit rate of user $u \in \mathcal{U}_s$
λ_u^{miss}	Cache miss rate of user $u \in \mathcal{U}_s$
γ_s	Percentage of backhaul resources for slice $s \in \mathcal{S}$
β_s	Percentage of cache resources for slice $s \in \mathcal{S}$
α_s	Variable that defines if slice $s \in \mathcal{S}$ is served or not
w_s	Cost of slice $s \in \mathcal{S}$
$\overline{D}_s$	Delay requirement of slice $s \in \mathcal{S}$
$D(\gamma_s, \beta_s)$	Achieved delay for slice $s \in \mathcal{S}$ given resources γ_s and β_s

and the SD-RAN controller. However, in some cases (semi-cooperative, cooperative slices) anonymous content statistics are shared to improve the network performance.

Each network slice contains a set of users $\mathcal{U}_s$ that can request files among the file catalog $\mathcal{K}$. The probability of a user $u \in \mathcal{U}_s$ to request file f is $p_{u,f}$ and follows a Zipf distribution [Vo+18]. For each file $f \in \mathcal{K}$, where the set size is $|\mathcal{K}|$, the probability is calculated according to

$$p_{u,f}^{z,|\mathcal{K}|} = \frac{1/(f^z)}{\sum_{j=1}^{|\mathcal{K}|} 1/j^z}. \tag{6.1}$$

The number of file requests per unit of time follows a Poisson distribution with rate $\lambda_u, u \in \mathcal{U}_s$. This profile is then utilized by the slice managers and the SD-RAN controller to help assigning the cache and backhaul resources. To achieve this procedure, the usage of cache and backhaul resources needs to be considered. In turn, this is enabled via modeling cache hit and miss rate. In the following section we introduce these models. Before moving on to the description of the next sections, we illustrate the list of variables and parameters used in this chapter in Table 6.1.

6.2.1 Caching Model

We adopt the caching technique introduced in [Vo+18], where for each file in the slice's catalog the hit probability is defined as:

$$h_{s,f} = \frac{\sum_{u=1}^{|\mathcal{U}_s|} \lambda_{u,f}}{\sum_{u=1}^{|\mathcal{U}_s|} \lambda_u} \cdot \beta_s \cdot B, \quad \forall\ s \in \mathcal{S}, \tag{6.2}$$

where $\lambda_{u,f} = p_{u,f} \cdot \lambda_u, u \in \mathcal{U}_s$ denotes the request rate of each file f for each user u in slice s. The rate of files that are found in the cache is referred to as the hit rate. For each user u, the hit rate is denoted by λ_u^{hit} and is expressed as $\lambda_u^{\text{hit}} = \sum_{f=1}^{|\mathcal{K}|} \lambda_{u,f} \cdot h_{s,f}$. Finally, the caching miss rate for each user is given by

$$\lambda_u^{\text{miss}} = \lambda_u - \lambda_u^{\text{hit}} = \lambda_u \{1 - \sum_{f=1}^{|\mathcal{K}|} p_{u,f} \cdot \frac{\sum_{u=1}^{|\mathcal{U}_s|} \lambda_{u,f}}{\sum_{u=1}^{|\mathcal{U}_s|} \lambda_u} \cdot \beta_s \cdot B\}. \tag{6.3}$$

As can be observed from Eq. (6.3), the larger the percentage of cache resources β_s that is assigned to a slice, the lower the caching miss rate.

6.2.2 Backhaul Model

In our system, once the file results in a cache hit, it is downloaded with no delay to the user in the aircraft. In contrast, if the file is not found in the cache, then the file should be downloaded from the data center on the ground. This includes the round trip delay from the satellite link. In that case, the backhaul link should be used. We model the delay of the backhaul link as an M/M/1 queuing system and the backhaul delay is:

$$\mathrm{D}(\gamma_s, \beta_s) = \begin{cases} \frac{1}{\mu_s(\gamma_s) - \lambda_s^{\text{miss}}(\beta_s)} + t_{rtt}, & 0 \leq \lambda_s^{\text{miss}}(\beta_s) < \mu_s(\gamma_s) \\ \infty, & \text{otherwise} \end{cases} \tag{6.4}$$

where $\mu_s(\gamma_s) = \gamma_s \cdot \frac{\Gamma}{L}$, is the backhaul rate in terms of files/s, where L is the size of each file in Mbit and Γ the capacity in Mbps. Finally, t_{rtt} is added as the satellite round trip time, whereas $\lambda_s^{\text{miss}} = \sum_{u=1}^{|\mathcal{U}_s|} \lambda_u^{\text{miss}}$ the total cache miss rate of the network slice.

6.2.3 Learning Model

As already described at the beginning of the section, the SD-RAN controller collects statistics from the slice managers that are willing to share their information with respect to their user profiles. In the case of cooperative slices, we assume that the SD-RAN controller possesses from the beginning all the information about the user profiles from the respective slice managers. Such details can be obtained by the slice managers by utilizing sophisticated estimation techniques before the flight or by requesting user preferences in advance. Alternatively, for semi-cooperative and uncooperative slices, the slice manager estimates the file profile for its users over the flight time according to a learning model, which is detailed in Alg. 4. The initial estimate for the user file profiles of each slice (i.e., $t = 0$) follows a uniform distribution. For uncooperative slices, the estimate does not change over time as no information is shared between the uncooperative slice managers and the SD-RAN controller. On the contrary, for each semi-cooperative slice, the slice manager keeps track of the user preferences at each time instance t and builds a file profile $p_{t,f}$ for that time instance. Moreover, it updates the estimated file profile $p_{s,f}$ with a learning rate α and sends these information to the SD-RAN controller to perform the resource allocation.

Algorithm 4 Slice File Profile Updates

```
1: Inputs: $|\mathcal{S}|$, $K$, $\alpha$
2: Initialization:
3: $p_{s,f} \leftarrow \frac{1}{|\mathcal{K}|}, \quad \forall f \in K, \forall s \in \mathcal{S}.$
4: for $t \in$ Learning Rounds do
5:     Slice Manager:
6:     Record the anonymous requested files for each user of the slice with respect to $\lambda_u$ and
       create $p_{t,f}$
7:     Update $p_{s,f} := p_{s,f} + \alpha \cdot p_{t,f}, \quad \forall f \in K$
8:     Send $p_{s,f}$ to the SD-RAN controller
9: end for
```

6.2.4 Optimization Formulation

The goal of the SD-RAN controller is to distribute the cache and backhaul resources in an efficient manner. This translates to maximizing the number of served slices in the network. To that end, we model our optimization as a resource maximization problem while satisfying the slice constraints. The developed optimization is denoted by $\mathcal{P}_0$ and is detailed in the following.

Let a_s be a binary decision variable whose value is 1 if slice $s \in \mathcal{S}$ is served and 0 otherwise. Let γ_s and β_s be continuous decision variables and denote the amount of cache and backhaul resources in % assigned to each slice s. Moreover, let $\overline{D}_s$ denote each slice's average tolerable delay requirement. The optimization problem is described as:

$$\mathcal{P}_0 : \max_{\gamma_s, \beta_s} \sum_s a_s \tag{6.5}$$

$$\text{s.t.} \quad \sum_{s \in S} \gamma_s \cdot a_s \leq 1, \tag{6.6}$$

$$\sum_{s \in S} \beta_s \cdot a_s \leq 1, \tag{6.7}$$

$$\lambda_s^{\text{miss}} \geq 0 \quad \forall s \in \mathcal{S}, \tag{6.8}$$

$$\mu_s - \lambda_s^{\text{miss}} \geq 0, \quad \forall s \in \mathcal{S}, \tag{6.9}$$

$$f_s = \begin{cases} 1, & \text{if } \lambda_s^{\text{miss}} > 0 \\ 0, & \text{otherwise} \end{cases} \tag{6.10}$$

$$\mathrm{D}(\gamma_s, \beta_s) \leq a_s \cdot f_s \cdot \overline{D}_s, \quad \forall s \in \mathcal{S}. \tag{6.11}$$

Constraints (6.6)-(6.7), ensure that the number of backhaul, and cache resources cannot exceed 100% for the assigned slices. Let constraints (6.8) and (6.9) assure that the cache miss rate can never be smaller than 0, whereas the cache miss rate cannot surpass the serving rate. Furthermore, let f_s be a binary variable that takes the value 1 if $\lambda_s^{\text{miss}} > 0$ and 0 otherwise as detailed in Eq. (6.10). This is an auxiliary variable used to solve $\mathcal{P}_0$. At last, constraint (6.11) guarantees that if the slice s receives any resource, then the average delay $\overline{D}_s$ requested by a slice $s \in \mathcal{S}$ must be fulfilled. If $f_s = 0$, the delay is 0 due to all the requests resulting in a cache hit, whereas if $f_s = 1$ the backhaul link must be utilized.

When identifying the number of served slices, from the solution we can obtain for each slice $s \in \mathcal{S}$, the minimum required amount of resources to fulfill its delay constraints. We refer to these amount of resources as the slice cost. In order to be fair among the slices and increase their chances of being selected, even for slices with high cost, we propose an alternative maximization problem similar to $\mathcal{P}_0$. The slice cost denoted by w_s is added as a price that has to be paid by the slice owner to increase its chances of being selected and name this problem $\mathcal{P}_1$ as:

$$\mathcal{P}_1 : \max_{\gamma_s, \beta_s} \sum_s a_s \cdot w_s. \tag{6.12}$$

The optimization problem $\mathcal{P}_1$ consists of the same constraints as $\mathcal{P}_0$.

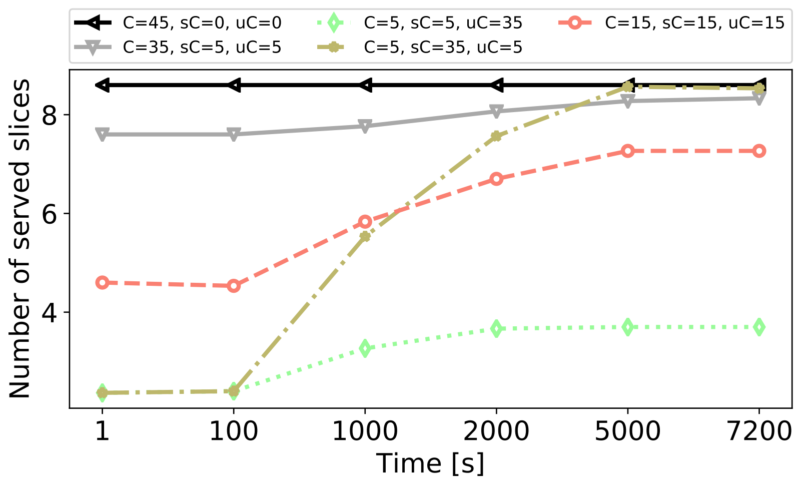


Figure 6.2: Number of maximum served slices with respect to time for various slice type combinations. Increasing time entails higher accuracy of the user profile for semi-cooperative slices.

6.3 Performance Evaluation

In the performance evaluation, we investigate the effect of slice cooperation in the overall system performance and demonstrate the results of our proposed optimization approach $\mathcal{P}_0$ stated in Section 6.2.4. We utilize ApOpt solver of Gekko [Bea18] in Python to solve our optimization. Initially, we categorize slices into types as cooperative (C), semi-cooperative (sC) and uncooperative (uC). Accordingly, we evaluate their cost depending on the system parameters cache size B, backhaul capacity Γ, slice delay requirement $\overline{D}_s$, and slice arrival rate λ_s. Finally, we make use of the defined slice cost and propose a pricing method via setting w_s, formulated in $\mathcal{P}_1$, for network slices in order to increase their chances of being served. Unless stated otherwise, we consider video streaming and web browsing as applications, $|\mathcal{K}|$ = 5000 files per slice, and the file size of L = 100 KB and a satellite round trip time of 200 ms.

6.3.1 Maximum Number of Served Slices

In this subsection, we investigate the impact of combinations of slice types in terms of the maximum number of served slices. For the given scenario, we assume 45 slices each with a distinct file profile following a Zipf distribution with a different z value ranging from 0 to 1. Moreover, we select a cache size of B = 2000 files [Vo+18], backhaul capacity of Γ = 112 Mbps (i.e., reflecting a real aircraft setting [Gog]), whereas for the slices we consider an average delay requirement of $\overline{D}_s$ = 1 s and a slice arrival rate of λ_s = 200 files/s consisting of 20 users each requesting on average 10 files/s. For our evaluation, we select 5 different configurations for the slice types. The first is the one where all the slices are cooperative. In the other three configurations, alternatively one of the slice types is the majority. The last

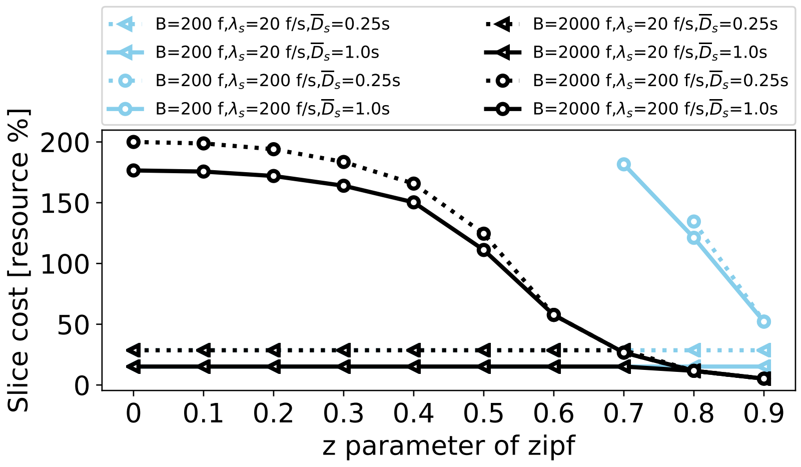


Figure 6.3: Cost of cooperative slices with respect to various delay, slice arrival and cache combinations for different Zipf distribution shape parameters z.

configuration contains an average number of slice types. For each of the configurations, we run the optimization 100 times generating different user file profiles each run. The results are illustrated in Fig. 6.2.

The scenario where all the slices are cooperative admits the highest number of slices. The algorithm minimizes the number of resources allocated per slice which is achieved easier if the user profile is known. At time 1 s, the configurations with most cooperative slices record higher number of slices served. However, with time the number of slices served increases also for configurations with semi-cooperative slices. At time 7200 s, i.e., when a good estimate of the user profile is obtained, the outcome of the configuration with semi-cooperative and cooperative slices is almost the same. The rationale behind this result lies in the fact that with time, the slice manager is able to predict the correct user profile for the slices. Finally, the configurations where the uncooperative slices are the majority demonstrates the least number of admitted slices. Thus, we can conclude that different slice types, even though they are homogeneous across all the parameters, incur different resource cost. In this example, 9 slices are served with cooperative slices compared to 3 with uncooperative slices. The effect of cooperation is 200% more slices served.

We again stress that the time axis in Fig. 6.2 can also be considered as the user profile estimation accuracy, as with the increase of time slice managers acquire more information with respect to the user profiles and therefore can accurately predict user requests. In the initial state (1 s), the slice managers possess an inaccurate estimate about the user profile, whereas in the last state (7200 s) an accurate estimate is obtained by the slice managers.

6.3.2 Slice Type Cost

In this subsection, we investigate the cost of a network slice for various slice types. The cost of a network slice is defined as the % of cache and backhaul resource usage and it is evaluated while varying the z parameter of the Zipf distribution. In our evaluations, eight different configurations are investigated with a high and low cache size $B \in \{200, 2000\}$ files, a high and low slice arrival rate $\lambda_s \in \{20, 200\}$ file requests per second and a high and low average delay requirement $\overline{D}_s \in \{0.25, 1\}$ seconds.

Fig. 6.3 illustrates the results for the cooperative slices, where the SD-RAN controller knows the file distribution perfectly from the beginning. The y-axis depicts the cost and the x-axis depicts the z parameter. The configurations that request more than the available network resources (i.e., 200%) are considered infeasible and thus are not presented in the figure.

In general, the slice cost increases with increasing slice arrival rate λ_s and decreasing average tolerable delay $\overline{D}_s$. Moreover, the slice cost is inversely proportional to the cache size B. According to the Zipf distribution, higher values of z lead to more skewed distributions. Therefore, the slice cost decreases with increasing z as less cache resources are needed to store the most popular files for the slices. In the low slice arrival rate scenario (i.e., $\lambda_s = 20$ files/s) presented by the triangular shape, using the cache is more costly than using the backhaul resources. Therefore, backhaul resources are always booked for that slice and as such, irrespective of the values of z and cache size, the cost is the same and the black and blue lines coincide. Yet, in a high cache size scenario depicted by black lines in Fig. 6.3, for more skewed file profiles (i.e., $z \geq 0.8$) the cache resources are less costly and we can observe an overall drop in the slice cost. Imposing tight delay constraints increases the resource cost only slightly. This is demonstrated with the differences between dotted and straight lines.

In a similar fashion, we show the cost of semi-cooperative slices in Fig. 6.4, where the SD-RAN controller has a semi-accurate file distribution of the slice. The semi-cooperative slice cost at time t=1000 s is shown in Fig. 6.4. A analogous trend with the cost of cooperative slices is observed. However, as opposed to the same configurations at time t=1000 s, the cost for semi-cooperative slices is higher. Nonetheless, the semi-cooperative slice cost at time t=5000 s is almost the same as that of cooperative slices due to increasing file profile accuracy. Therefore, those results are omitted. Furthermore, the slice cost at time t=1s is the same as that of the uncooperative slices as elaborated next.

The cost of the uncooperative slices can be obtained from Fig. 6.3, when observing the results for $z = 0$, which denotes a uniform distribution. The delay requirement makes a slight difference in terms of cost. Increasing the arrival rate by 10× almost quadruples the cost. Cache size is the most dominant factor and the slice cannot be served with low cache size. It

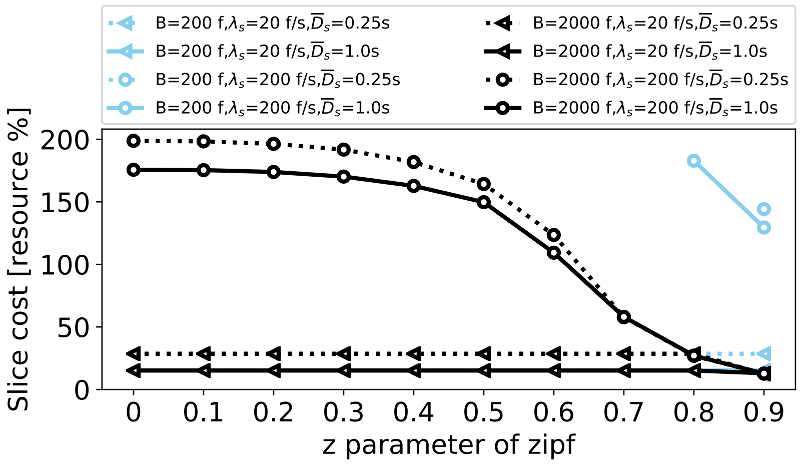
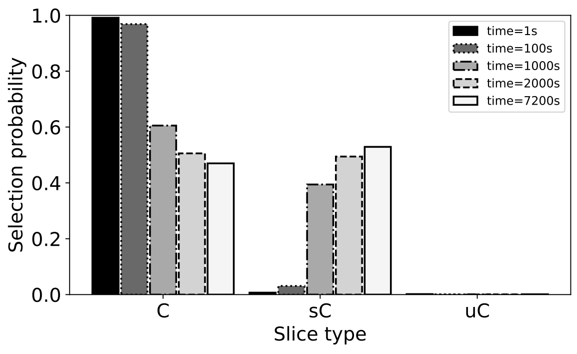
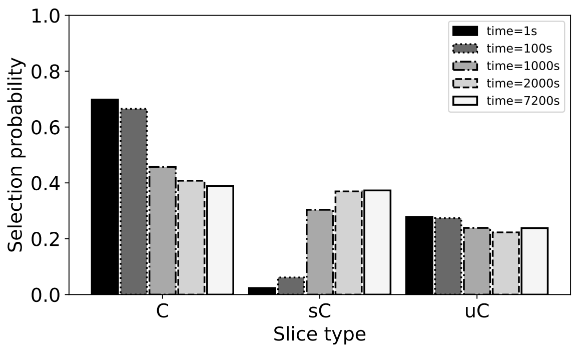


Figure 6.4: Cost of semi-cooperative slices with respect to various delay, slice arrival and cache combinations for different Zipf distribution shape parameters z at t=1000 s.



(a) No weights



(b) Weights for all slices

Figure 6.5: Probability of selection for multiple time instances and slice types: cooperative (C), semi-cooperative (sC) and uncooperative(uC). If weights are introduced into the objective function, slices have almost equal chances of being served.

can be concluded that the highest cost is noted for uncooperative slices since no information is shared with the SD-RAN controller to improve the estimation.

6.3.3 Probability of Slice Selection

In this subsection, we investigate the slice discrimination with respect to the slice type. We achieve this by investigating the slice selection probability for various scenarios. We consider the case of 45 network slices while randomly selecting the slice types and performing 100 optimization rounds. We want to emphasize that the resources are scarce, i.e., not sufficient to serve all slices. The z parameter is randomly selected for each slice at each round. This problem is inputted in $\mathcal{P}_0$ and the average number of served slices is taken over all cases and the selection probability is reflected in Fig. 6.5a. The x-axis represents the slice type, whereas the y-axis shows the selection probability. The results reflect that in a resource scarce scenario the uncooperative slices are almost never served and semi-cooperative slices are served only after an accurate estimation of the user profile.

In order to overcome such starvation for different slice types, we envision that the SD-RAN controller should use weights w_s as in $\mathcal{P}_1$ for the resource allocation problem. Furthermore, we propose to set these weights as the slice costs introduced in the previous section. We have illustrated the selection probabilities with $\mathcal{P}_1$ in Fig. 6.5b. In the optimal case we observe $\approx 33\%$ selection probability for all the slice types, which is almost achieved by the proposed weights. These weights are considered as the price to pay for providing fairness to each slice without sacrificing network resources.

6.4 Emerging 6G Edge Network Use Cases

As mentioned at the beginning of this chapter, the current terrestrial and satellite network integration is not sufficient to provide the required level of QoS for 5G and beyond networks. Therefore, as pointed out in [Hua+19], a middle layer between the satellite and terrestrial layers, namely the aeronautical layer is required in order to provide improved resource sharing and high spectral efficiency. Consequently, civil aircrafts, Unmanned Aerial Vehicles (UAV)s and High-Altitude Platform (HAP)s will be utilized in combination with mm-wave links to provide high bandwidth. This concept envisions a three-dimensional network architecture, which includes terrestrial (RANs), non-terrestrial (satellites) and aeronautical networks as elaborated in [Aza+21; Dao+21]. This can therefore lead to the introduction of novel use cases for next generation networks such as for instance the introduction of aeronautical edge computing as proposed in [Man+22; Pap+23b].

Considering such a three-dimensional setup, the results presented in this chapter as well as the system model and architecture design can be considered as initial points that enhance the aeronautical network layer by the usage of SDN and network slicing concepts already available in the other two network layers. For instance, the RAN tackled in the previous chapter of this thesis as well as satellite networks as described in [Pap+18; Pap+20a; Ber+15; Fer+16; Liu+18].

6.5 Summary

In this chapter, in contrast with the previous chapters of this thesis, we focused on an multi-network environment considering not only terrestrial networks, but also satellites and aeronautical components. In that regard, we developed a system architecture based on the SDN and network slicing logic. We further investigated the network slicing problem for an IFECS-based system, where both cache and backhaul resources are considered as bottlenecks.

We followed a realistic scenario, assuming that there is no perfect knowledge of the user profiles, and we provided a new definition for network slices with respect to their willingness to share user statistics. They were grouped as cooperative, semi-cooperative and uncooperative slices. Moreover, we proposed a MINLP approach that incorporates all slice requirements and aims at maximizing the number of served slices in the network. For each slice type we defined the deployment cost, and investigated its dependency on time, system and slice parameters. In order to increase the chances of uncooperative slices being served, we proposed a pricing system added to the original maximization problem. Our proposed solution increases the selection probability of uncooperative slices by 33% if the slice cost is paid by the slice owners. Overall, we showed that cooperative slicing can revolutionize the IFECS system as it accommodates 200% more services compared to uncooperative slicing.

Finally, given the strict latency requirements for specific service types in 5G, we highlighted the necessity of a three-dimensional architecture that includes not only the traditional satellite and terrestrial network layers, but also an additional aeronautical layer.

Chapter 7

Conclusion

SDN and network slicing are envisioned as promising tools to accommodate the wide range of heterogeneous applications in 5G networks, especially at the RAN side. However, while the 5GC and NR functions are well defined in 3GPP, novel concepts such as SD-RAN still remain unspecified in the standard. Therefore, the integration of methods that leverage both concepts of SD-RAN and RAN slicing comprise a set of challenges, which concern mainly the establishment of effective architectural concepts and the development of optimization approaches that improve the QoS requirements of users in RAN.

In order to develop efficient RAN slicing solutions with the use of SD-RAN, initially, a clear understanding of the newly introduced SD-RAN concepts and SD-RAN controllers is necessary. Thus, measurements and evaluation of existing SD-RAN platforms and their controllers is of utmost importance. Moreover, the investigation of the effect of various design and architectural principles of SD-RAN approaches is critical, as they demystify important trade-offs with respect to users' QoS.

Obtaining such valuable details fosters the design of effective optimization approaches, which becomes necessary in order to further improve the network performance, in particular with respect to users in the network. Yet, for RAN slicing, specific characteristics of wireless channel environments need to be considered to establish optimal solutions. Such details concern knowledge of stochastic wireless channels and interference management techniques.

Finally, to enable network ubiquity, which is one of the main requirements of 5G, the attention has to be drawn beyond RAN. That includes the introduction of other networks apart from cellular ones, like satellite and aeronautical networks. However, optimal solutions that leverage such a multi-network setup are challenging and require further investigation.

7.1 Summary

This thesis addresses four main research questions with respect to the concept of RAN softwarization and virtualization, listed as follows:

- **R1:** Definition, architecture design and optimization algorithms for RAN slicing.
- **R2:** Measurements with SD-RAN controller platforms, analysis of SD-RAN control plane design and performance evaluation of the user QoS.
- **R3:** Proposal of a system architecture for enabling network slicing in multi-network environments such as cellular, satellite and aeronautical.
- **R4:** Integration of SD-RAN into the existing 5G network infrastructure.

Regarding research questions **R1** and **R4**, in Chapter 3, this thesis provided a system model for RAN slicing based on the concepts of programmability and virtualization through the use of the SD-RAN paradigm. Considering both single and multi-BS scenarios, approaches based on the Lyapunov optimization were proposed to satisfy slices' QoS and maintain slice isolation. For the latter, this thesis also proposed a clear definition comparing proposals from the state of the art. The main message of Chapter 3 is that leveraging the knowledge of wireless channel characteristics, a dynamic re-assignment of wireless resources improves the system throughput compared to state-of-the-art approaches that only consider static methods. Another important outcome from this chapter was the integration of the developed system in the 5G architecture, where the provisioning of SD-RAN as a 5G function in the context of RAN slicing was suggested.

To maintain user performance guarantees in SD-RAN environments, in Chapter 4, this thesis proposed `MARC`, a novel benchmarking tool for SD-RAN architectures and controllers. Using `MARC`, measurements and performance evaluations were provided for `FlexRAN` and `5G-EmPOWER`, which are state-of-the-art SD-RAN platforms. This led to obtaining important details with respect to the stable operational region of the considered SD-RAN controllers.

To mitigate the control plane inconsistencies that were observed when using `MARC` for centralized SD-RAN environments, Chapter 5 proposed a new system, called `Delphi`. Through the use of `Delphi`, investigations on the impact of the SD-RAN control plane imperfections on the data plane using both mathematical models and measurement tools were conducted. Furthermore, Chapter 5 provided a 5G-based SD-RAN simulator with measurements from `MARC`. In that regard, evaluations with respect to the effect of the SD-RAN control plane design on the user QoS performance for various metrics, such as user throughput, latency and packet drop rate was provided. Results demonstrated that a single SD-RAN controller cannot handle a large number of devices in the data plane as it degrades the performance of the users by 20% in terms of throughput, 140% in terms of average delay and 5× in terms of packet loss ratio, which can be mitigated with a distributed control plane. Both Chapter 4 and Chapter 5 dealt with research question **R2**.

Regarding research question **R3**, Chapter 6 of this thesis focused on a multi-network environment considering not only terrestrial networks, but also satellites and aeronautical components. To that end, a system architecture based on the principles of SDN and network slicing was developed. Taking the IFECS as an example, an optimization approach that maximizes the number of served network slices in an aircraft was proposed and solved.

7.2 Future Work

This thesis provided important foundations in the concepts of programmability and virtualization of next generation RANs. However, there exist other topics of research towards a mature establishment of SD-RAN concepts in 5G and emerging 6G networks. Some of the following potential avenues of future work are of particular interest:

Integration of network slicing optimization approaches in realistic system setups. While Chapter 3 and Chapter 6 of this thesis presented valuable optimization approaches in the context of RAN and multi-network slicing, still the solutions were evaluated on simulation environments. However, the integration of such algorithms in realistic system setups is an interesting avenue of research. First, such an integration provides validation on important concepts such as slice isolation from the software and hardware perspective, which is difficult to be captured realistically by simulation tools. Further, the implementation in realistic setups often reveals important system characteristics and assumptions that do not hold in simulations. Thus, a better assessment can be provided.

Implementation of a distributed SD-RAN control plane architecture and performance evaluation. In Chapter 5 of this thesis, a 5G-enabled SD-RAN simulator was provided given measurements with realistic SD-RAN controllers. Yet, such a simulation setup still cannot grasp completely the challenges that arise from real distributed control plane implementations, such as the consistency of SD-RAN controller databases, as well as software-related complications with respect to the transfer, reading and writing of databases during a control handover. In that regard, the research community would benefit from the implementation and evaluation of such a distributed control plane design.

Development of mathematical models and performance methods for various user QoS requirements in SD-RAN environments. Following the measurements provided by `Delphi` and the pertaining mathematical models in Chapter 5, various alternative user QoS metrics, like packet delay and packet loss ratio can be derived. Moreover, with the development of a distributed SD-RAN control plane setup, the performance evaluation provided by `Delphi` for a centralized SD-RAN approach with a single SD-RAN controller can be extended and studied for more controllers as well. That evaluation could accurately assess the trade-offs between a distributed and centralized control plane for SD-RAN environments.

Design and optimization of a three-dimensional network architecture. To support the requirement for ubiquity of 5G networks, in Chapter 6, the necessity of a three-dimensional network architecture, consisting of terrestrial, non-terrestrial and aeronautical networks was highlighted. Hence, the system design for the integration of the aforementioned networks becomes an interesting research area. Thus, further evaluations of the benefits that network slicing and SDN bring can be performed. Moreover, approaches that include MEC to reduce the delay for applications in the sky like IFECS could be of particular interest. The consideration of MEC applications, in turn leads to additional research directions concerning efficient MEC placement and optimal routing for applications in the sky.

Bibliography

Publications by the author

Journal publications

[Aya+20b] O. Ayan, H. M. Gürsu, A. Papa, and W. Kellerer. “Probability analysis of age of information in multi-hop networks.” In: *IEEE Networking Letters* 2.2 (2020), pp. 76–80. DOI: 10.1109/LNET.2020.2976991.

[Man+22] J. von Mankowski, E. Durmaz, A. Papa, H. Vijayaraghavan, and W. Kellerer. “Aerial-Aided Multi-Access Edge Computing: Dynamic and Joint Optimization of Task and Service Placement and Routing in Multi-Layer Networks.” In: *IEEE Transactions on Aerospace and Electronic Systems* Early Access (2022). DOI: 10.1109/TAES.2022.3217430.

[Pap+20a] A. Papa, T. De Cola, P. Vizarreta, M. He, C. Mas-Machuca, and W. Kellerer. “Design and evaluation of reconfigurable SDN LEO constellations.” In: *IEEE Transactions on Network and Service Management* 17.3 (2020), pp. 1432–1445. DOI: 10.1109/TNSM.2020.2993400.

[Pap+20b] A. Papa, R. Durner, L. Goratti, T. Rasheed, and W. Kellerer. “Controlling Next-Generation Software-Defined RANs.” In: *IEEE Communications Magazine* (2020), pp. 58–64. DOI: 10.1109/MCOM.001.1900732.

[Pap+21a] A. Papa, R. Durner, E. Goshi, L. Goratti, T. Rasheed, A. Blenk, and W. Kellerer. “MARC: On Modeling and Analysis of Software-Defined Radio Access Network Controllers.” In: *IEEE Transactions on Network and Service Management* 18.4 (2021), pp. 4602–4615. DOI: 10.1109/TNSM.2021.3095673.

[Pap+21c] A. Papa, A. Jano, S. Ayvaşık, O. Ayan, H. M. Gürsu, and W. Kellerer. “User-Based Quality of Service Aware Multi-Cell Radio Access Network Slicing.” In: *IEEE Transactions on Network and Service Management* (2021). DOI: 10.1109/TNSM.2021.3122230.

[Pap+23a] A. Papa, P. Kutsevol, F. Mehmeti, and W. Kellerer. “Delphi: Computing the Maximum Achievable Throughput in SD-RAN Environments.” In: *IEEE Transactions on Network and Service Management (under revision)*. 2023.

[Pap+23b] A. Papa, J. von Mankowski, H. Vijayaraghavan, B. Mafakheri, L. Goratti, and W. Kellerer. “Enabling 6G Applications in the Sky: Aeronautical Federation Framework.” In: *IEEE Network* Early Access (2023). DOI: 10.1109/MNET.132.2200526.

Conference publications

[Cal+20] R. M. Calvo, T. de Cola, J. Poliak, L. Macri, A. Papa, et al. “Optical Feeder Links for Future Very High-Throughput Satellite Systems in B5G Networks.” In: *Proc. of the IEEE European Conference on Optical Communications (ECOC)*. 2020, pp. 1–4. DOI: 10.1109/ECOC48923.2020.9333405.

[MPK23] F. Mehmeti, A. Papa, and W. Kellerer. “Maximizing Network Throughput Using SD-RAN.” In: *Proc. of the IEEE Consumer Communications & Networking Conference (CCNC)*. 2023. DOI: 10.1109/CCNC51644.2023.10059833.

[Pap+18] A. Papa, T. De Cola, P. Vizarreta, M. He, C. M. Machuca, and W. Kellerer. “Dynamic SDN controller placement in a LEO constellation satellite network.” In: *Proc. of the IEEE Global Communications Conference (GLOBECOM)*. 2018, pp. 206–212. DOI: 10.1109/GLOCOM.2018.8647843.

[Pap+19a] A. Papa, R. Durner, F. Edinger, and W. Kellerer. “SDRBench: A Software-Defined Radio Access Network Controller Benchmark.” In: *Proc. of the IEEE Conference on Network Softwarization (NetSoft Workshops)*. 2019, pp. 36–41. DOI: 10.1109/NETSOFT.2019.8806697.

[Pap+19b] A. Papa, M. Klugel, L. Goratti, T. Rasheed, and W. Kellerer. “Optimizing dynamic RAN slicing in programmable 5G networks.” In: *Proc. of the IEEE International Conference on Communications (ICC)*. 2019, pp. 1–7. DOI: 10.1109/ICC.2019.8761163.

[Pap+21b] A. Papa, H. M. Gürsu, L. Goratti, T. Rasheed, and W. Kellerer. “Cost of Network Slice Collaboration: Edge Network Slicing for In-Flight Connectivity.” In: *Proc. of the IEEE International Conference on Communications (ICC)*. 2021, pp. 1–6. DOI: 10.1109/ICC42927.2021.9500820.

[Pap+22] A. Papa, P. Kutsevol, F. Mehmeti, and W. Kellerer. "Effects of SD-RAN Control Plane Design on User Quality of Service." In: *Proc. of the IEEE Conference on Network Softwarization (NetSoft)*. 2022. DOI: 10.1109/NetSoft54395.2022.9844029.

[UPK22] R.-M. Ursu, A. Papa, and W. Kellerer. "Experimental Evaluation of Downlink Scheduling Algorithms using OpenAirInterface." In: *Proc. of the IEEE Wireless Communications and Networking Conference Workshops (WCNCW)*. 2022. DOI: 10.1109/WCNC51071.2022.9771597.

Web

[Pap21] A. Papa. *Source code, configuration files, and data sets associated to MARC*. 2021. URL: https://github.com/arled-papa/marc.

General publications

[3GPa] 3GPP. *TR 21.915 V1.1.0 (2019-03); Technical Report; Summary of Rel-15 Work Items (Release 15)*.

[3GPb] 3GPP. *TR 23.711 V14.0.0 (2016-09); Enhancements of Dedicated Core Networks selection mechanism (Release 14)*.

[3GPc] 3GPP. *TR 28.801 V15.1.0 (2018-01); Technical Report; Study on management and orchestration of network slicing for next generation network (Release 15)*.

[3GPd] 3GPP. *TS 38.401 V15.6.0 (2019-07); Technical Specification Group Radio Access Network; Architecture description (Release 15)*.

[3GP08] 3GPP. *Evolved Universal Terrestrial Radio Access (E-UTRA); User Equipment (UE) radio transmission and reception*. Technical Report (TR) 36.803. 3rd Generation Partnership Project (3GPP), 2008. URL: https://portal.3gpp.org/desktopmodules/Specifications/SpecificationDetails.aspx?specificationId=2482.

[3GP17a] 3GPP. *NR; Base Station (BS) radio transmission and reception*. Technical Specification (TS) 38.104. 3rd Generation Partnership Project (3GPP), Mar. 2017.

[3GP17b] 3GPP. *NR; Multiplexing and channel coding*. Technical Specification (TS) 38.212. 3rd Generation Partnership Project (3GPP), Apr. 2017.

[3GP17c] 3GPP. *NR; Physical layer procedures for data*. Technical Specification (TS) 38.214. 3rd Generation Partnership Project (3GPP), Apr. 2017.

[3GP17d] 3GPP. *NR; User Equipment (UE) radio access capabilities*. Technical Specification (TS) 38.306. 3rd Generation Partnership Project (3GPP), Mar. 2017.

[3GP18a] 3GPP. *NG Radio Access Network (NG-RAN); Stage 2 functional specification of User Equipment (UE) positioning in NG-RAN*. Technical Specification (TS) 38.305. 3rd Generation Partnership Project (3GPP), June 2018.

[3GP18b] 3GPP. *NR; NR and NG-RAN Overall description; Stage-2*. Technical Specification (TS) 38.300. 3rd Generation Partnership Project (3GPP), Jan. 2018.

[3GP19a] 3GPP. *Study on NR positioning support*. Technical Specification (TS) 38.855. 3rd Generation Partnership Project (3GPP), Mar. 2019.

[3GP19b] 3GPP. *TS 22.261 V16.8.0 (2019-06); Technical Specification Group Service and System Aspects; Service requirements for the 5G systems (Release 16)*. 2019.

[Afo+18] I. Afolabi, T. Taleb, K. Samdanis, A. Ksentini, and H. Flinck. "Network slicing and softwarization: A survey on principles, enabling technologies, and solutions." In: *IEEE Communications Surveys & Tutorials* 20.3 (2018), pp. 2429–2453.

[AGK19] S. Ayvaşık, H. M. Gürsu, and W. Kellerer. "Veni Vidi Dixi: Reliable wireless communication with depth images." In: *Proc. of the 15th International Conference on Emerging Networking Experiments And Technologies (CoNEXT)*. 2019, pp. 172–185.

[AJK20] A. M. Alba, S. Janardhanan, and W. Kellerer. "Dynamics of the flexible functional split selection in 5G networks." In: *Proc. of the IEEE Global Communications Conference (GLOBECOM)*. IEEE. 2020, pp. 1–6.

[AK19] A. M. Alba and W. Kellerer. "A dynamic functional split in 5G radio access networks." In: *Proc. of the IEEE Global Communications Conference (GLOBECOM)*. IEEE. 2019, pp. 1–6.

[All16] Alliance, NGMN. "Description of network slicing concept." In: *NGMN 5G P* 1.1 (2016), pp. 1–11.

[AÖK21] O. Ayan, H. Y. Özkan, and W. Kellerer. "An experimental framework for Age of Information and networked control via Software-Defined Radios." In: *Proc. of the IEEE International Conference on Communications (ICC)*. 2021, pp. 1–6.

[AP19] H. D. R. Albonda and J. Pérez-Romero. “An efficient RAN slicing strategy for a heterogeneous network with eMBB and V2X services.” In: *IEEE access* 7 (2019), pp. 44771–44782.

[ASR15] M. Y. Arslan, K. Sundaresan, and S. Rangarajan. “Software-Defined Networking in cellular Radio Access Networks: potential and challenges.” In: *IEEE Communications Magazine* 53.1 (2015), pp. 150–156.

[AVK19] A. M. Alba, J. H. G. Velásquez, and W. Kellerer. “An adaptive functional split in 5G networks.” In: *Proc. of the IEEE INFOCOM Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. 2019, pp. 410–416.

[AWL15] I. F. Akyildiz, P. Wang, and S.-C. Lin. “SoftAir: A Software-Defined Networking architecture for 5G wireless systems.” In: *Computer Networks* 85 (2015), pp. 1–18.

[Aya+19] J. A. Ayala-Romero, A. Garcia-Saavedra, M. Gramaglia, X. Costa-Perez, A. Banchs, and J. J. Alcaraz. “vrAIn: A Deep Learning Approach Tailoring Computing and Radio Resources in Virtualized RANs.” In: *Proc. of the 25th Annual International Conference on Mobile Computing and Networking (MobiCom)*. 2019, pp. 1–16.

[Aya+20a] J. A. Ayala-Romero, A. Garcia-Saavedra, M. Gramaglia, X. Costa-Perez, A. Banchs, and J. J. Alcaraz. “vrain: Deep learning based orchestration for computing and radio resources in vRANs.” In: *IEEE Transactions on Mobile Computing* (2020).

[Aza+21] M. M. Azari, S. Solanki, S. Chatzinotas, O. Kodheli, H. Sallouha, et al. “Evolution of Non-Terrestrial Networks From 5G to 6G: A Survey.” In: *:2107.06881* (2021).

[BBD14] E. Bastug, M. Bennis, and M. Debbah. “Living on the edge: The role of proactive caching in 5G wireless networks.” In: *IEEE Communications Magazine* 52.8 (2014), pp. 82–89.

[Bea18] Beal, Logan and Hill, Daniel and Martin, R and Hedengren, John. “GEKKO Optimization Suite.” In: *Processes* 6.8 (2018), p. 106. DOI: 10.3390/pr6080106.

[Beg+17] D. Bega, M. Gramaglia, A. Banchs, V. Sciancalepore, K. Samdanis, and X. Costa-Perez. “Optimising 5G infrastructure markets: The business of network slicing.” In: *Proc. of the IEEE Conference on Computer Communications (INFOCOM)*. 2017, pp. 1–9.

[Beg+18] D. Bega, A. Banchs, M. Gramaglia, X. Costa-Pérez, and P. Rost. "CARES: Computation-aware scheduling in virtualized radio access networks." In: *IEEE Transactions on Wireless Communications* 17.12 (2018), pp. 7993–8006.

[Ber+14a] P. Berde, M. Gerola, J. Hart, Y. Higuchi, M. Kobayashi, et al. "ONOS: towards an open, distributed SDN OS." In: *Proc. of the third workshop on Hot topics in Software Defined Networking (HotSDN)*. 2014, pp. 1–6.

[Ber+14b] C. J. Bernardos, A. De La Oliva, P. Serrano, A. Banchs, L. M. Contreras, H. Jin, and J. C. Zúñiga. "An Architecture for Software-Defined Wireless Networking." In: *IEEE Wireless Communications* 21.3 (2014), pp. 52–61.

[Ber+15] L. Bertaux, S. Medjiah, P. Berthou, S. Abdellatif, A. Hakiri, et al. "Software-Defined Networking and Virtualization for Broadband Satellite Networks." In: *IEEE Communications Magazine* 53.3 (2015), pp. 54–60.

[Ber+19] L. Bertizzolo, L. Bonati, E. Demirors, and T. Melodia. "Arena: A 64-antenna SDR-based ceiling grid testbed for sub-6 GHz radio spectrum research." In: *Proc. of the 13th International Workshop on Wireless Network Testbeds, Experimental Evaluation & Characterization*. 2019, pp. 5–12.

[Ble+15] A. Blenk, A. Basta, M. Reisslein, and W. Kellerer. "Survey on network virtualization hypervisors for Software-Defined Networking." In: *IEEE Communications Surveys & Tutorials* 18.1 (2015), pp. 655–685.

[Ble+16] A. Blenk, A. Basta, J. Zerwas, M. Reisslein, and W. Kellerer. "Control plane latency with SDN network hypervisors: The cost of virtualization." In: *IEEE Transactions on Network and Service Management* 13.3 (2016), pp. 366–380.

[Ble+19] A. Blenk, A. Basta, W. Kellerer, and S. Schmid. "On the impact of the network hypervisor on virtual network performance." In: *Proc. of the IEEE/IFIP Networking Conference*. 2019, pp. 1–9.

[Boe] Boeing. *The Boeing, 2007*. URL: http://www.boeing.com.

[Boe+18] L. Boero, R. Bruschi, F. Davoli, M. Marchese, and F. Patrone. "Satellite networking integration in the 5G ecosystem: Research trends and open challenges." In: *IEEE Network* 32.5 (2018), pp. 9–15.

[Bon+21a] L. Bonati, S. D'Oro, S. Basagni, and T. Melodia. "SCOPE: an open and softwarized prototyping platform for NextG systems." In: *Proc. of the 19th Annual International Conference on Mobile Systems, Applications, and Services*. 2021, pp. 415–426.

[Bon+21b] L. Bonati, P. Johari, M. Polese, S. D'Oro, S. Mohanti, et al. "Colosseum: Large-Scale Wireless Experimentation Through Hardware-in-the-Loop Network Emulation." In: *Proc. of the IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*. 2021, pp. 105–113.

[Bre+21] J. Breen, A. Buffmire, J. Duerig, K. Dutt, E. Eide, et al. "POWDER: Platform for open wireless data-driven experimental research." In: *Computer Networks* 197 (2021), p. 108281.

[Bro16] G. Brown. "Exploring 5G new radio: Use cases capabilities & timeline." In: *Qualcomm White Paper* (2016), pp. 1–12.

[Cab+17] P. Caballero, A. Banchs, G. De Veciana, and X. Costa-Pérez. "Multi-tenant radio access network slicing: Statistical multiplexing of spatial loads." In: *IEEE/ACM Transactions on Networking* 25.5 (2017), pp. 3044–3058.

[Cab+19] P. Caballero, A. Banchs, G. De Veciana, and X. Costa-Pérez. "Network slicing games: Enabling customization in multi-tenant mobile networks." In: *IEEE/ACM Transactions on Networking* 27.2 (2019), pp. 662–675.

[Cap+12] 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." In: *IEEE communications surveys & tutorials* 15.2 (2012), pp. 678–700.

[Che+19] D. A. Chekired, M. A. Togou, L. Khoukhi, and A. Ksentini. "5G-slicing-enabled scalable SDN core network: Toward an ultra-low latency of autonomous driving service." In: *IEEE Journal on Selected Areas in Communications* 37.8 (2019), pp. 1769–1782.

[Chr19] Christopher Wallace. *Bringing 5G networks indoors*. 2019. URL: https://www.ericsson.com/en/reports-and-papers/white-papers/bringing-5g-networks-indoors.

[CKR19] E. Coronado, S. N. Khan, and R. Riggio. "5G-EmPOWER: A Software-Defined Networking platform for 5G radio access networks." In: *IEEE Transactions on Network and Service Management* 16.2 (2019), pp. 715–728.

[CNS18] C.-Y. Chang, N. Nikaein, and T. Spyropoulos. "Radio access network resource slicing for flexible service execution." In: *Proc. of the IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*. 2018, pp. 668–673.

[Cos+18] S. Costanzo, I. Fajjari, N. Aitsaadi, and R. Langar. “A network slicing prototype for a flexible cloud radio access network.” In: *Proc. of the 15th IEEE Annual Consumer Communications & Networking Conference*. 2018, pp. 1–4.

[CR19] E. Coronado and R. Riggio. “Flow-based network slicing: Mapping the future mobile radio access networks.” In: *Proc. of the IEEE International Conference on Computer Communication and Networks (ICCCN)*. 2019, pp. 1–9.

[CYR20] E. Coronado, Z. Yousaf, and R. Riggio. “LightEdge: mapping the evolution of Multi-Access Edge computing in cellular networks.” In: *IEEE Communications Magazine* 58.4 (2020), pp. 24–30.

[CYY16] L. Cui, F. R. Yu, and Q. Yan. “When big data meets Software-Defined Networking: SDN for big data and big data for SDN.” In: *IEEE network* 30.1 (2016), pp. 58–65.

[Dah18] Dahlman, Erik and Parkvall, Stefan and Skold, Johan. *5G NR: The next generation wireless access technology*. Academic Press, 2018.

[Dan+19] K. R. Dandekar, S. Begashaw, M. Jacovic, A. Lackpour, I. Rasheed, et al. “Grid Software-Defined Radio Network testbed for hybrid measurement and emulation.” In: *Proc. of the 16th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*. 2019, pp. 1–9.

[Dao+21] N.-N. Dao, Q.-V. Pham, N. H. Tu, T. T. Thanh, V. N. Q. Bao, D. S. Lakew, and S. Cho. “Survey on Aerial Radio Access Networks: Toward a Comprehensive 6G Access Infrastructure.” In: *IEEE Communications Surveys & Tutorials* 23.2 (2021), pp. 1193–1225.

[Dix+] A. Dixit, F. Hao, S. Mukherjee, T. Lakshman, and R. R. Kompella. “ElastiCon; an elastic distributed SDN controller.” In: *Proc. of the 2014 ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS)*, pp. 17–27.

[DOr+19] S. D’Oro, F. Restuccia, A. Talamonti, and T. Melodia. “The slice is served: Enforcing radio access network slicing in virtualized 5G systems.” In: *Proc. of the IEEE Conference on Computer Communications (INFOCOM)*. 2019, pp. 442–450.

[DOr+20] S. D’Oro, L. Bonati, F. Restuccia, M. Polese, M. Zorzi, and T. Melodia. “Sl-EDGE: Network slicing at the edge.” In: *Proc. of the Twenty-First International Symposium on Theory, Algorithmic Foundations, and Protocol Design for Mobile Networks and Mobile Computing*. 2020, pp. 1–10.

[ETS] ETSI. *TS 23.501 V15.2.0 (2018-06); Technical Specification; System Architecture for the 5G System*.

[ETS18] ETSI. *5G NR Overall description: 3GPP TS 38.300 version 15.3.1 Release 15*. www.etsi.org. Technical specification. 2018.

[Ett] Ettus Research. *USRP B210*. URL: https://www.ettus.com/all-products/ub210-kit/.

[Eur] Eurecom. *oai cn*. https://gitlab.eurecom.fr/mosaic5g/mosaic5g/-/wikis/tutorials/oai-cn.

[Fed] Federal Communications Commission (FCC). *Spectrum Crunch*. URL: https://www.fcc.gov/engineering-technology/policy-and-rules-division/general/radio-spectrum-allocation.

[Fer+16] R. Ferrús, H. Koumaras, O. Sallent, G. Agapiou, T. Rasheed, et al. "SDN/NFV-enabled satellite communications networks: Opportunities, scenarios and challenges." In: *Physical Communication* 18 (2016), pp. 95–112.

[Fer+18] R. Ferrus, O. Sallent, J. Pérez-Romero, and R. Agusti. "On 5G radio access network slicing: Radio interface protocol features and configuration." In: *IEEE Communications Magazine* 56.5 (2018), pp. 184–192.

[FL02] H. Fattah and C. Leung. "An overview of scheduling algorithms in wireless multimedia networks." In: *IEEE wireless communications* 9.5 (2002), pp. 76–83.

[FMK17] X. Foukas, M. K. Marina, and K. Kontovasilis. "Orion: RAN slicing for a flexible and cost-effective multi-service mobile network architecture." In: *Proc. of the 23rd annual International Conference On Mobile Computing And Networking (MobiCom)*. 2017, pp. 127–140.

[Fou] O. N. Foundation. *Libfluid*. URL: https://opennetworkingfoundation.github.io/libfluid/md%5C_doc%5C_Benchmarks.html.

[Fou+16] X. Foukas, N. Nikaein, M. M. Kassem, M. K. Marina, and K. Kontovasilis. "FlexRAN: A flexible and programmable platform for Software-Defined Radio Access Networks." In: *Proc. of the 12th ACM International Conference on emerging Networking EXperiments and Technologies (CoNEXT)*. 2016, pp. 427–441.

[Fou+17] X. Foukas, G. Patounas, A. Elmokashfi, and M. K. Marina. "Network slicing in 5G: Survey and challenges." In: *IEEE Communications Magazine* 55.5 (2017), pp. 94–100.

[FR21] X. Foukas and B. Radunovic. “Concordia: teaching the 5G vRAN to share compute.” In: *Proc. of the 2021 ACM SIGCOMM 2021 Conference*. 2021, pp. 580–596.

[Gar+18a] G. Garcia-Aviles, M. Gramaglia, P. Serrano, and A. Banchs. “POSENS: A practical open source solution for end-to-end network slicing.” In: *IEEE Wireless Communications* 25.5 (2018), pp. 30–37.

[Gar+18b] A. Garcia-Saavedra, J. X. Salvat, X. Li, and X. Costa-Perez. “WizHaul: On the centralization degree of cloud RAN next generation fronthaul.” In: *IEEE Transactions on Mobile Computing* 17.10 (2018), pp. 2452–2466.

[Gar+21] G. Garcia-Aviles, A. Garcia-Saavedra, M. Gramaglia, X. Costa-Perez, P. Serrano, and A. Banchs. “Nuberu: Reliable RAN virtualization in shared platforms.” In: *Proc. of the 27th Annual International Conference on Mobile Computing and Networking (MobiCom)*. 2021, pp. 749–761.

[Gin+21] D. Ginthör, R. Guillaume, M. Schüngel, and H. D. Schotten. “5G RAN Slicing for Deterministic Traffic.” In: *Proc. of the IEEE Wireless Communications and Networking Conference (WCNC)*. 2021, pp. 1–6.

[GKP18] G. Giambene, S. Kota, and P. Pillai. “Satellite-5G integration: A network perspective.” In: *IEEE Network* 32.5 (2018), pp. 25–31.

[Gog] "Gogoair". *"How do you get loads of bandwidth to a plane"*. URL: https://www.gogoair.com/learning-center/get-loads-bandwidth-plane/.

[Gom+16] I. Gomez-Miguelez, A. Garcia-Saavedra, P. D. Sutton, P. Serrano, C. Cano, and D. J. Leith. “srsLTE: An open-source platform for LTE evolution and experimentation.” In: *Proc. of the Tenth ACM International Workshop on Wireless Network Testbeds, Experimental Evaluation, and Characterization*. 2016, pp. 25–32.

[Goo19] Google. *Protocol Buffers*. 2019. URL: https://developers.google.com/protocol-buffers/.

[Gri+18] F. Gringoli, P. Patras, C. Donato, P. Serrano, and Y. Grunenberger. “Performance assessment of open software platforms for 5G prototyping.” In: *IEEE Wireless Communications* 25.5 (2018), pp. 10–15.

[Gud+13] A. Gudipati, D. Perry, L. E. Li, and S. Katti. “SoftRAN: Software-Defined Radio Access Network.” In: *Proc. of the second ACM SIGCOMM workshop on Hot topics in Software Defined Networking (HotSDN)*. 2013, pp. 25–30.

[Gup+19] R. Gupta, S. Tanwar, S. Tyagi, and N. Kumar. “Tactile internet and its applications in 5G era: A comprehensive review.” In: *International Journal of Communication Systems* 32.14 (2019), e3981.

[Har+20] D. Harutyunyan, N. Shahriar, R. Boutaba, and R. Riggio. “Latency and mobility-aware service function chain placement in 5G networks.” In: *IEEE Transactions on Mobile Computing* (2020).

[HLS18] B. Han, J. Lianghai, and H. D. Schotten. “Slice as an evolutionary service: Genetic optimization for inter-slice resource management in 5G networks.” In: *IEEE Access* 6 (2018), pp. 33137–33147.

[Hos+14] E. Hossain, M. Rasti, H. Tabassum, and A. Abdelnasser. “Evolution toward 5G multi-tier cellular wireless networks: An interference management perspective.” In: *IEEE Wireless Communications* 21.3 (2014), pp. 118–127.

[Hua+19] X. Huang, J. A. Zhang, R. P. Liu, Y. J. Guo, and L. Hanzo. “Airplane-Aided Integrated Networking for 6G Wireless: Will it work?” In: *IEEE Vehicular Technology Magazine* 14.3 (2019), pp. 84–91.

[Hua16] Huawei. *Small cell network*. 2016. URL: https://www.huawei.com/minisite/hwmbbf16/insights/small_cell_solution.pdf.

[Inm] "Inmarsat". *"The European Aviation Network"*. URL: https://www.telekom.com/static/-/288318/2/150921-product-sheet-si.

[ITU] ITU-R. *Minimum requirements related to technical performance for imt-2020 radio interface(s), Report itu-r m.2410-0.*

[Jar+12] M. Jarschel, F. Lehrieder, Z. Magyari, and R. Pries. “A flexible OpenFlow-controller benchmark.” In: *Proc. of the IEEE European Workshop on Software Defined Networking*. 2012, pp. 48–53.

[Jar+14] M. Jarschel, T. Zinner, T. Hoßfeld, P. Tran-Gia, and W. Kellerer. “Interfaces, attributes, and use cases: A compass for SDN.” In: *IEEE Communications Magazine* 52.6 (2014), pp. 210–217.

[Jin+13] X. Jin, L. E. Li, L. Vanbever, and J. Rexford. “Softcell: Scalable and flexible cellular core network architecture.” In: *Proc. of the ninth ACM Conference on emerging Networking EXperiments and Technologies (CoNEXT)*. 2013, pp. 163–174.

[Jun+14] V. Jungnickel, K. Manolakis, W. Zirwas, B. Panzner, V. Braun, et al. “The role of small cells, coordinated multipoint, and massive MIMO in 5G.” In: *IEEE Communications Magazine* 52.5 (2014), pp. 44–51.

[KD17] M. Karakus and A. Durresi. “A survey: Control plane scalability issues and approaches in software-defined networking (SDN).” In: *Computer Networks* 112 (2017), pp. 279–293.

[Kel+19] W. Kellerer, P. Kalmbach, A. Blenk, A. Basta, M. Reisslein, and S. Schmid. “Adaptable and data-driven softwarized networks: Review, opportunities, and challenges.” In: *Proc. of the IEEE* 107.4 (2019), pp. 711–731.

[KGD18] J. Krolikowski, A. Giovanidis, and M. Di Renzo. “Optimal cache leasing from a mobile network operator to a content provider.” In: *Proc. of the IEEE Conference on Computer Communications (INFOCOM)*. 2018.

[KL14] S. Katti and L. E. Li. “Radiovisor: A slicing plane for radio access networks.” In: *Proc. of the Open Networking Summit 2014*. 2014.

[KLG] M. I. Kamel, L. B. Le, and A. Girard. “LTE wireless network virtualization: Dynamic slicing via flexible scheduling.” In: *Vehicular Technology Conference (VTC Fall), 2014 IEEE 80th*, pp. 1–5.

[KN17] A. Ksentini and N. Nikaein. “Toward enforcing network slicing on RAN: Flexibility and resources abstraction.” In: *IEEE Communications Magazine* 55.6 (2017), pp. 102–108.

[Kok+11] R. Kokku, R. Mahindra, H. Zhang, and S. Rangarajan. “NVS: A substrate for virtualizing wireless resources in cellular networks.” In: *IEEE/ACM Transactions on Networking* 20.5 (2011), pp. 1333–1346.

[Kou+19] K. Koutlia, R. Ferrús, E. Coronado, R. Riggio, F. Casadevall, A. Umbert, and J. Pérez-Romero. “Design and experimental validation of a software-defined radio access network testbed with slicing support.” In: *Wireless Communications and Mobile Computing* (2019).

[Kre+14] D. Kreutz, F. M. Ramos, P. E. Verissimo, C. E. Rothenberg, S. Azodolmolky, and S. Uhlig. “Software-Defined Networking: A comprehensive survey.” In: *Proc. of the IEEE* 103.1 (2014), pp. 14–76.

[KS18] A. T. Z. Kasgari and W. Saad. “Stochastic optimization and control framework for 5G network slicing with effective isolation.” In: *Proc. of the 52nd Annual Conference on Information Sciences and Systems (CISS)*. 2018, pp. 1–6.

[KT19] S. Kukliński and L. Tomaszewski. “Key Performance Indicators for 5G network slicing.” In: *Proc. of the IEEE Conference on Network Softwarization (NetSoft)*. 2019, pp. 464–471.

[KW15] G. Ku and J. M. Walsh. “Resource Allocation and Link Adaptation in LTE and LTE Advanced: A Tutorial.” In: *IEEE Communications Surveys & Tutorials* 17.3 (2015).

[LEH20] E. Liu, E. Effiok, and J. Hitchcock. “Survey on health care applications in 5G networks.” In: *IET Communications* 14.7 (2020), pp. 1073–1080.

[Li+20a] J. Li, B. Lei, N. Li, and H. Lv. “A Load Balancing Approach for Distributed SDN Architecture Based on Sharing Data Store.” In: *Proc. of the IEEE 21st Asia-Pacific Network Operations and Management Symposium (APNOMS)*. 2020, pp. 31–36.

[Li+20b] J. Li, W. Shi, P. Yang, Q. Ye, X. S. Shen, X. Li, and J. Rao. “A hierarchical soft RAN slicing framework for differentiated service provisioning.” In: *IEEE Wireless Communications* 27.6 (2020), pp. 90–97.

[Liu+18] J. Liu, Y. Shi, L. Zhao, Y. Cao, W. Sun, and N. Kato. “Joint placement of controllers and gateways in SDN-enabled 5G-satellite integrated network.” In: *IEEE Journal on Selected Areas in Communications* 36.2 (2018), pp. 221–232.

[LJ16] G. Liu and D. Jiang. “5G: Vision and requirements for mobile communication system towards year 2020.” In: *Chinese Journal of Engineering* 2016.2016 (2016), p. 8.

[LY15] C. Liang and F. R. Yu. “Wireless virtualization for next generation mobile cellular networks.” In: *IEEE wireless communications* 22.1 (2015), pp. 61–69.

[Man+19] S. Mandelli, M. Andrews, S. Borst, and S. Klein. “Satisfying network slicing constraints via 5G MAC scheduling.” In: *Proc. of the IEEE Conference on Computer Communications (INFOCOM)*. 2019, pp. 2332–2340.

[Mar] Markets and Markets. *"In-flight Entertainment and Connectivity Market"*. URL: https://www.marketsandmarkets.com/Market-Reports/in-flight-entertainment-communications-market-860.html.

[Mar+16] P. Marsch, I. Da Silva, O. Bulakci, M. Tesanovic, S. E. El Ayoubi, et al. “5G Radio Access Network Architecture: Design guidelines and key considerations.” In: *IEEE Communications Magazine* 54.11 (2016), pp. 24–32.

[Mar+18] C. Marquez, M. Gramaglia, M. Fiore, A. Banchs, and X. Costa-Perez. “How should I slice my network? A multi-service empirical evaluation of resource sharing efficiency.” In: *Proc. of the 24th Annual International Conference on Mobile Computing and Networking (MobiCom)*. 2018, pp. 191–206.

[MC08] N. Moraitis and P. Constantinou. "Radio channel measurements and characterization inside aircrafts for in-cabin wireless networks." In: *Proc. of the IEEE 68th Vehicular Technology Conference*. 2008, pp. 1–5.

[McK+08] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, et al. "OpenFlow: enabling innovation in campus networks." In: *ACM SIGCOMM computer communication review* 38.2 (2008), pp. 69–74.

[Med+14] J. Medved, R. Varga, A. Tkacik, and K. Gray. "Opendaylight: Towards a model-driven sdn controller architecture." In: *Proc. of the IEEE International Symposium on a World of Wireless, Mobile and Multimedia Networks 2014*. 2014, pp. 1–6.

[Mei+21] J. Mei, X. Wang, K. Zheng, G. Boudreau, A. B. Sediq, and H. Abou-zeid. "Intelligent Radio Access Network Slicing for Service Provisioning in 6G: A Hierarchical Deep Reinforcement Learning Approach." In: *IEEE Transactions on Communications* (2021).

[MLG06] H. G. Myung, J. Lim, and D. J. Goodman. "Single carrier FDMA for uplink wireless transmission." In: *IEEE Vehicular Technology Magazine* 1.3 (2006), pp. 30–38.

[Moh+15] A. Mohamed, O. Onireti, M. A. Imran, A. Imran, and R. Tafazolli. "Control-data separation architecture for cellular radio access networks: A survey and outlook." In: *IEEE Communications Surveys & Tutorials* 18.1 (2015), pp. 446–465.

[Mor+] N. Moraitis, P. Constantinou, F. Perez Fontan, and P. Valtr. "Propagation measurements and comparison with EM techniques for in-cabin wireless networks." In: *EURASIP Journal on Wireless Communications and Networking* 2009 (), pp. 1–13.

[Mosa] Mosaic5G. *flexran-rtc*. URL: https://gitlab.eurecom.fr/flexran/flexran-rtc.

[Mosb] Mosaic5G. *l2 sim*. https://gitlab.eurecom.fr/mosaic5g/mosaic5g/-/wikis/tutorials/l2-sim.

[Mosc] Mosaic5G. *mosaic5g-oai-sim*. https://gitlab.eurecom.fr/oai/openairinterface5g/-/tree/mosaic5g-oai-sim.

[Mos20] Mosaic5G. *FlexRAN northbound API documentation*. 2020. URL: https://mosaic5g.io/apidocs/flexran/#api-Stats-SetStatsConf.

[Mos21] Mosaic5G. *FlexRAN*. 2021. URL: https://mosaic5g.io/flexran/.

[MP21] F. Mehmeti and T. L. Porta. “Analyzing a 5G Dataset and Modeling Metrics of Interest.” In: *Proc. of the IEEE MSN*. 2021.

[MR19] F. Mehmeti and C. Rosenberg. “How expensive is consistency? Performance analysis of consistent rate provisioning to mobile users in cellular networks.” In: *IEEE Transactions on Mobile Computing* 18.5 (2019).

[Nam+14] W. Nam, D. Bai, J. Lee, and I. Kang. “Advanced interference management for 5G cellular networks.” In: *IEEE Communications Magazine* 52.5 (2014), pp. 52–60.

[Nee10] Neely, Michael J. “Stochastic network optimization with application to communication and queueing systems.” In: *Synthesis Lectures on Communication Networks* 3.1 (2010), pp. 1–211.

[Nik+14] N. Nikaein, M. K. Marina, S. Manickam, A. Dawson, R. Knopp, and C. Bonnet. “OpenAirInterface: A flexible platform for 5G research.” In: *ACM SIGCOMM Computer Communication Review* 44.5 (2014), pp. 33–38.

[Nok17] Nokia. *Small cells and femtocells*. 2017. URL: https://www.nokia.com/networks/portfolio/small-cells/#small-cells.

[NVH18] N. Nikaein, X. Vasilakos, and A. Huang. “LL-MEC: Enabling low latency edge applications.” In: *Proc. of the IEEE International Conference on Cloud Networking (CloudNet)*. 2018, pp. 1–7.

[O-Ra] O-RAN Working Group 1. *O-RAN-WG1-O-RAN Architecture Description-v04.00.00, O-RAN.WG1.O-RAN-Architecture-Description-v04.00, Technical Specification, March 2021.*

[O-Rb] O-RAN Working Group 2. *O-RAN A1 interface: Application Protocol 3.0, O-RAN.WG2.A1AP-v03.01, Technical Specification, March 2021.*

[O-Rc] O-RAN Working Group 2. *O-RAN Non-RT RIC Architecture 1.0, O-RAN.WG2.Non-RT-RIC-ARCH-TS-v01.00, Technical Specification, July, 2021.*

[O-Rd] O-RAN Working Group 3. *O-RAN Near-Real-time RAN Intelligent Controller Architecture and E2 General Aspects and Principles 2.00, O-RAN.WG3.E2GAP-v02.01, Technical Specification, July, 2021.*

[O-Re] O-RAN Working Group 3. *O-RAN Near-Real-time RAN Intelligent Controller Near-RT RIC Architecture 2.00, O-RAN.WG3.RICARCH-v02.00, Technical Specification, March, 2021.*

[Oug+19] E. J. Oughton, Z. Frias, S. van der Gaast, and R. van der Berg. “Assessing the capacity, coverage and cost of 5G infrastructure strategies: Analysis of the Netherlands.” In: *Telematics and Informatics* 37 (2019), pp. 50–69.

[Pop+18] P. Popovski, K. F. Trillingsgaard, O. Simeone, and G. Durisi. “5G wireless network slicing for eMBB, URLLC, and mMTC: A communication-theoretic view.” In: *IEEE Access* 6 (2018), pp. 55765–55779.

[Pyt19a] Python. *Asynchronous I/O*. 2019. URL: https://docs.python.org/3/library/asyncio.html.

[Pyt19b] Python. *Construct libary*. 2019. URL: https://pypi.org/project/construct/.

[Pyt19c] Python. *psutil libary*. 2019. URL: https://pypi.org/project/psutil/.

[Qaz+17] Z. A. Qazi, M. Walls, A. Panda, V. Sekar, S. Ratnasamy, and S. Shenker. “A high performance packet core for next generation cellular networks.” In: *Proc. of the Conference of the ACM Special Interest Group on Data Communication*. 2017, pp. 348–361.

[Rac+20] D. Raca, D. Leahy, C. J. Sreenan, and J. J. Quinlan. “Beyond throughput, the next generation: a 5G dataset with channel and context metrics.” In: *Proc. of the 11th ACM multimedia systems conference*. 2020, pp. 303–308.

[Rak20] Rakuten. *How elegant software can make 5G networks more resilient*. 2020. URL: https://rakuten.today/blog/5g-network-reliability-lightreading.html.

[Ram+18] K. Ramantas, A. Antonopoulos, E. Kartsakli, P.-V. Mekikis, J. Vardakas, and C. Verikoukis. “A C-RAN based 5G platform with a fully virtualized, SDN controlled optical/wireless fronthaul.” In: *Proc. of the IEEE 20th International Conference on Transparent Optical Networks*. 2018, pp. 1–4.

[Rig+14] R. Riggio, K. Gomez, L. Goratti, R. Fedrizzi, and T. Rasheed. “V-Cell: Going beyond the cell abstraction in 5G mobile networks.” In: *Proc. of the IEEE Network Operations and Management Symposium (NOMS)*. 2014, pp. 1–5.

[Rig+21] R. Riggio, E. Coronado, N. Linder, A. Jovanka, G. Mastinu, et al. “AI@ EDGE: A Secure and Reusable Artificial Intelligence Platform for Edge Computing.” In: *2021 Joint European Conference on Networks and Communications & 6G Summit (EuCNC/6G Summit)*. 2021, pp. 610–615.

[Rob19] "Roberto Riggio". *Empower Runtime*. 2019. URL: https://github.com/5g-empower/empower-runtime.

[Ros+17a] P. Rost, C. Mannweiler, D. S. Michalopoulos, C. Sartori, V. Sciancalepore, et al. “Network slicing to enable scalability and flexibility in 5G mobile networks.” In: *IEEE Communications Magazine* 55.5 (2017), pp. 72–79.

[Ros+17b] A. Rostami, P. Ohlen, K. Wang, Z. Ghebretensae, B. Skubic, M. Santos, and A. Vidal. “Orchestration of RAN and transport networks for 5G: An SDN approach.” In: *IEEE Communications Magazine* 55.4 (2017), pp. 64–70.

[RS11] A. Ribeiro and R. C. Sofia. *A survey on mobility models for wireless networks*. 2011.

[Ryu] Ryu. *RYU SDN Framework*. URL: https://osrg.github.io/ryu-book/en/html/.

[Sal+17] O. Sallent, J. Perez-Romero, R. Ferrus, and R. Agusti. “On radio access network slicing from a radio resource management perspective.” In: *IEEE Wireless Communications* 24.5 (2017), pp. 166–174.

[Sal+18] J. X. Salvat, L. Zanzi, A. Garcia-Saavedra, V. Sciancalepore, and X. Costa-Perez. “Overbooking network slices through yield-driven end-to-end orchestration.” In: *Proc. of the 14th International Conference on emerging Networking EXperiments and Technologies (CoNEXT)*. 2018, pp. 353–365.

[SCB19] V. Sciancalepore, X. Costa-Perez, and A. Banchs. “RL-NSB: Reinforcement Learning-Based 5G Network Slice Broker.” In: *IEEE/ACM Transactions on Networking* 27.4 (2019), pp. 1543–1557.

[SCN19] R. Schmidt, C.-Y. Chang, and N. Nikaein. “Slice Scheduling with QoS-Guarantee Towards 5G.” In: *Proc. of the IEEE Global Communications Conference (GLOBECOM)*. 2019, pp. 1–7.

[SCS16] K. Samdanis, X. Costa-Perez, and V. Sciancalepore. “From network sharing to multi-tenancy: The 5G network slice broker.” In: *IEEE Communications Magazine* 54.7 (2016), pp. 32–39.

[SDC19] V. Sciancalepore, M. Di Renzo, and X. Costa-Perez. “STORNS: Stochastic radio access network slicing.” In: *Proc. of the IEEE International Conference on Communications (ICC)*. 2019, pp. 1–7.

[Ser15] Series, M. “IMT Vision–Framework and overall objectives of the future development of IMT for 2020 and beyond.” In: *Recommendation ITU* 2083 (2015), p. 21.

[Sez+13] S. Sezer, S. Scott-Hayward, P. K. Chouhan, B. Fraser, D. Lake, et al. “Are we ready for SDN? Implementation challenges for Software-Defined Networks.” In: *IEEE Communications Magazine* 51.7 (2013), pp. 36–43.

[She] "Sherwood, R. and Yap, K.-K." *CBench Controller Benchmarker*. URL: http://www.openflowswitch.org/wk/index.php/Oflops,%202011.

[Shi+18] Q. Shi, L. Zhao, Y. Zhang, G. Zheng, F. R. Yu, and H.-H. Chen. "Energy-efficiency versus delay tradeoff in wireless networks virtualization." In: *IEEE Transactions on Vehicular Technology* 67.1 (2018), pp. 837–841.

[Shi+21] W. Shi, J. Li, P. Yang, Q. Ye, W. Zhuang, S. Shen, and X. Li. "Two-level Soft RAN Slicing for Customized Services in 5G-and-beyond Wireless Communications." In: *IEEE Transactions on Industrial Informatics* (2021).

[SIN21] R. Schmidt, M. Irazabal, and N. Nikaein. "FlexRIC: an SDK for next-generation SD-RANs." In: *Proc. of the 17th International Conference on emerging Networking EXperiments and Technologies (CoNEXT)*. 2021, pp. 411–425.

[SK17] E. Sakic and W. Kellerer. "Response time and availability study of RAFT consensus in distributed SDN control plane." In: *IEEE Transactions on Network and Service Management* 15.1 (2017), pp. 304–318.

[SM19] D. Sattar and A. Matrawy. "Optimal slice allocation in 5G core networks." In: *IEEE Networking Letters* 1.2 (2019), pp. 48–51.

[Sun+20] Y. Sun, S. Qin, G. Feng, L. Zhang, and M. Imran. "Service provisioning framework for RAN slicing: user admissibility, slice association and bandwidth allocation." In: *IEEE Transactions on Mobile Computing* (2020).

[Tal+17] T. Taleb, B. Mada, M.-I. Corici, A. Nakao, and H. Flinck. "PERMIT: Network slicing for personalized 5G mobile telecommunications." In: *IEEE Communications Magazine* 55.5 (2017), pp. 88–93.

[TG10] A. Tootoonchian and Y. Ganjali. "Hyperflow: A distributed control plane for openflow." In: *Proc. of the 2010 internet network management conference on Research on enterprise networking*. Vol. 3. 2010.

[TL18] T. D. Tran and L. B. Le. "Joint wireless access-backhaul network slicing and content caching optimization." In: *Proc. of the IEEE International Conference on Communications Workshops (ICC Workshops)*. 2018, pp. 1–6.

[Tri+18] R. Trivisonno, M. Condoluci, X. An, and T. Mahmoodi. "mIoT slice for 5G systems: Design and performance evaluation." In: *Sensors* 18.2 (2018), p. 635.

[Val20] D. Vallis. *5G breaks from proprietary systems, embraces open source RANs*. https://www.5gtechnologyworld.com/5g-breaks-from-proprietary-systems-embraces-open-source-rans/. 2020.

[Var+19] A. Varasteh, S. Hofmann, N. Deric, M. He, D. Schupke, W. Kellerer, and C. M. Machuca. “Mobility-aware joint service placement and routing in space-air-ground integrated networks.” In: *Proc. of the IEEE International Conference on Communications (ICC)*. 2019, pp. 1–7.

[Vil+20] I. Vilà, J. Pérez-Romero, O. Sallent, and A. Umbert. “Characterization of Radio Access Network slicing scenarios with 5G QoS provisioning.” In: *IEEE access* 8 (2020), pp. 51414–51430.

[Vo+18] P. L. Vo, M. N. Nguyen, T. A. Le, and N. H. Tran. “Slicing the edge: Resource allocation for RAN network slicing.” In: *IEEE Wireless Communications Letters* 7.6 (2018), pp. 970–973.

[Wan+14] X. Wang, M. Chen, T. Taleb, A. Ksentini, and V. C. Leung. “Cache in the air: Exploiting content caching and delivery techniques for 5G systems.” In: *IEEE Communications Magazine* 52.2 (2014), pp. 131–139.

[WGT19] Y. Wang, Y. Gu, and X. Tao. “Edge Network Slicing With Statistical QoS Provisioning.” In: *IEEE Wireless Communications Letters* 8.5 (2019), pp. 1464–1467.

[WL21] S. Wijethilaka and M. Liyanage. “Survey on network slicing for Internet of Things realization in 5G networks.” In: *IEEE Communications Surveys & Tutorials* 23.2 (2021), pp. 957–994.

[XCZ18] J. Xu, L. Chen, and P. Zhou. “Joint service caching and task offloading for mobile edge computing in dense networks.” In: *Proc. of the IEEE Conference on Computer Communications (INFOCOM)*. 2018, pp. 207–215.

[XE13] D. Xue and E. Ekici. “Delay-guaranteed cross-layer scheduling in multihop wireless networks.” In: *IEEE/ACM Transactions on Networking* 21.6 (2013), pp. 1696–1707.

[Xia+14] W. Xia, Y. Wen, C. H. Foh, D. Niyato, and H. Xie. “A survey on Software-Defined Networking.” In: *IEEE Communications Surveys & Tutorials* 17.1 (2014), pp. 27–51.

[Xu+16] F. Xu, H. Yao, C. Zhao, and C. Qiu. “Towards next generation Software-Defined Radio Access Network–architecture, deployment, and use case.” In: *EURASIP Journal on Wireless Communications and Networking* 2016.1 (2016), pp. 1–12.

[Xu+19] Y. Xu, M. Cello, I.-C. Wang, A. Walid, G. Wilfong, et al. “Dynamic switch migration in distributed Software-Defined Networks to achieve controller load balance.” In: *IEEE Journal on Selected Areas in Communications* 37.3 (2019), pp. 515–529.

[Yan+13] M. Yang, Y. Li, D. Jin, L. Su, S. Ma, and L. Zeng. “OpenRAN: a Software-Defined RAN architecture via virtualization.” In: *ACM SIGCOMM computer communication review*. Vol. 43. 4. 2013, pp. 549–550.

[Yan+18] P. Yang, N. Zhang, S. Zhang, L. Yu, J. Zhang, and X. S. Shen. “Content popularity prediction towards location-aware mobile edge caching.” In: *IEEE Transactions on Multimedia* 21.4 (2018), pp. 915–929.

[YKS14] V. Yazıcı, U. C. Kozat, and M. O. Sunay. “A new control plane for 5G network architecture with a case study on unified handoff, mobility, and routing management.” In: *IEEE Communications Magazine* 52.11 (2014), pp. 76–85.

[YQC17] L. Yin, L. Qiu, and Z. Chen. “Throughput-Maximum Resource Provision in the OFDMA-Based Wireless Virtual Network.” In: *Vehicular Technology Conference (VTC Spring)*. IEEE. 2017, pp. 1–6.

[Zha+17] H. Zhang, N. Liu, X. Chu, K. Long, A.-H. Aghvami, and V. C. Leung. “Network slicing based 5G and future mobile networks: mobility, resource management, and challenges.” In: *IEEE Communications Magazine* 55.8 (2017), pp. 138–145.

[Zha19] S. Zhang. “An overview of network slicing for 5G.” In: *IEEE Wireless Communications* 26.3 (2019), pp. 111–117.

List of Figures

List of Tables