UNIVERSITAT POLITÈCNICA
DE CATALUNYA
BARCELONATECH

PhD program in Signal Theory and Communications

Contribution to the modelling and evaluation of radio network slicing solutions in 5G

Doctoral thesis by:
Irene Vilà Muñoz

Thesis advisor:
Dr. Oriol Sallent Roig, Dra. Anna Umbert Juliana

Signal Theory and Communications Department
Barcelona, February 2022

To my beloved family

for their unconditional love and support.

A la meva família

pel seu amor i suport incondicional.

Abstract

Network slicing is a key feature of 5G architecture that allows the partitioning of the network into multiple logical networks, known as network slices, where each of them is customised according to the specific needs of a service or application. Thus, network slicing allows the materialisation of multi-tenant networks, in which a common network infrastructure is shared among multiple communication providers, acting as tenants and each of them using a different network slice. The support of multi-tenancy through slicing in the Radio Access Network (RAN), known as RAN slicing, is particularly challenging because it involves the configuration and operation of multiple and diverse RAN behaviours over the common pool of radio resources available at each of the RAN nodes. Moreover, this configuration needs to be performed in such a way that the specific requirements of each tenant are satisfied and, at the same time, the available radio resources are efficiently used. Therefore, new functionalities that allow the deployment of RAN slices are needed to be introduced at different levels, ranging from Radio Resource Management (RRM) functionalities that incorporate RAN slicing parameters to functionalities that support the lifecycle management of RAN slices. This thesis has addressed this need by proposing, developing and assessing diverse solutions for the support RAN slicing, which has allowed evaluating the capacities, requirements and limitations of network slicing in the RAN from diverse perspectives.

Specifically, this thesis is firstly focused on the analytical assessment of RRM functionalities that support multi-tenant and multi-services scenarios, where services are defined according to their 5G QoS requirements. This assessment is conducted through the Markov modelling of admission control policies and the statistical modelling of the resource allocation, both supporting multiple tenants and multiple services. Secondly, the thesis addresses the problem of slice admission control by proposing a methodology for the estimation of the radio resources required by a RAN slice based on data analytics. This methodology supports the decision on the admission or rejection of new RAN slice creation requests. Thirdly, the thesis explores the potential of artificial intelligence, and specifically, of Deep Reinforcement Learning (DRL) to deal with the capacity sharing problem in RAN slicing scenarios. To this end, a DRL-based capacity sharing solution that distributes the available capacity of a multi-cell scenario among multiple tenants is proposed and assessed. The solution consists in a Multi-Agent Reinforcement Learning (MARL) approach based on Deep Q-Network. Finally, this thesis discuses diverse implementation aspects of the DRL-based capacity sharing solution, including considerations on its compatibility with the standards, the impact of the training on the achieved performance, as well as the tools and technologies required for the deployment of the solution in the real network environment.

Resumen

El *Network Slicing* es una tecnología clave de la arquitectura del 5G que permite dividir la red en múltiples redes lógicas, conocidas como *network slices,* que se configuran de acuerdo a las necesidades de servicios y aplicaciones específicas. Así, el network slicing permite la materialización de las redes con múltiples inquilinos, donde una infraestructura de red común se comparte entre diferentes proveedores de comunicaciones, que actúan como inquilinos y que usan network slices diferentes. El soporte de múltiples inquilinos mediante el uso del network slicing en la red de acceso radio (RAN), que se conoce como RAN slicing, es un gran reto tecnológico, ya que comporta la configuración y operación de múltiples y diversos comportamientos sobre los recursos radio disponibles en cada uno de los nodos de la red de acceso. Además, esta configuración debe realizarse de tal manera que los requisitos específicos de cada inquilino se satisfagan y, al mismo tiempo, los recursos radio disponibles se utilicen eficazmente. Por lo tanto, es necesario introducir nuevas funcionalidades a diferentes niveles que permitan el despliegue de las RAN slices, desde funcionalidades relacionadas con la gestión de recursos radio (RRM) que incorporen parámetros para el RAN slicing a funcionalidades que proporcionen soporte a la gestión del ciclo de vida de las RAN slices. Esta tesis ha abordado esta necesidad proponiendo, desarrollando y evaluando diversas soluciones para el soporte del RAN slicing, lo que ha permitido analizar las capacidades, requisitos y limitaciones del RAN slicing desde diversas perspectivas.

Específicamente, esta tesis se centra, en primer lugar, en realizar un análisis de funcionalidades de RRM que soportan escenarios con múltiples inquilinos y múltiples servicios, donde los servicios se definen según sus requisitos de 5G QoS. Este análisis se lleva a cabo mediante la caracterización de políticas de control de admisión mediante un modelo de Markov y el modelado a nivel estadístico de la asignación de recursos, ambos soportando múltiples inquilinos y múltiples servicios. En segundo lugar, la tesis aborda el problema del control de admisión de network slices proponiendo una metodología para la estimación de los recursos radio requeridos por una RAN slice que se basa en análisis de datos. Esta metodología da soporte a la decisión sobre la admisión o el rechazo de nuevas solicitudes de creación de RAN slice. En tercer lugar, la tesis explora el potencial de la inteligencia artificial, y en concreto, de las técnicas de Deep Reinforcement Learning (DRL) para tratar el problema de compartición de capacidad en escenarios de RAN slicing. Para ello, se propone y evalúa una solución de compartición de capacidad basada en DRL que distribuye la capacidad disponible de un escenario multicelular entre múltiples inquilinos. Esta solución se plantea como una solución de Multi-Agent Reinforcement Learning (MARL) basado en Deep Q-Network. Finalmente, en esta tesis se tratan diversos aspectos relacionados con la implementación de la solución de reparto de

capacidad basada en DRL, incluyendo consideraciones sobre su compatibilidad con los estándares, el impacto del entrenamiento en el comportamiento y rendimiento conseguido, así como las herramientas y tecnologías necesarias para su despliegue en un entorno de red real.

Resum

El *Network Slicing* és una tecnologia clau de l'arquitectura del 5G que permet dividir la xarxa en múltiples xarxes lògiques, conegudes com a *network slices*, on cada una es configura d'acord a les necessitats d'un servei o aplicació específic. Així, el network slicing permet la materialització de les xarxes amb múltiples inquilins, on una infraestructura de xarxa comuna es comparteix entre diferents proveïdors de comunicacions, que actuen com a inquilins i utilitzen network slices diferents. El suport de múltiples inquilins mitjançant l'ús del network slicing a la xarxa d'accés ràdio (RAN), que es coneix com a RAN slicing, és un gran repte tecnològic, ja que comporta la configuració i operació de múltiples i diversos comportaments sobre els recursos ràdio disponibles a cadascun dels nodes de la xarxa d'accés. A més a més, aquesta configuració s'ha de portar a terme de forma que els requisits específics de cada inquilí es satisfacin i, al mateix temps, els recursos ràdio disponibles s'utilitzin eficientment. Per tant, és necessari introduir noves funcionalitats a diferents nivells que permetin el desplegament de les RAN slices, des de funcionalitats relacionades amb la gestió dels recursos ràdio (RRM) que incorporin paràmetres per al RAN slicing a funcionalitats que proporcionin suport a la gestió del cicle de vida de les RAN slices. Aquesta tesi ha adreçat aquesta necessitat proposant, desenvolupant i avaluant diverses solucions pel suport del RAN slicing, que han permès analitzar les capacitats, requisits i limitacions del RAN slicing des de diferents perspectives.

Específicament, aquesta tesi es centra, en primer lloc, en realitzar una anàlisi de les funcionalitats de RRM que suporten escenaris amb múltiples inquilins i múltiples serveis, on els serveis es defineixen d'acord amb els seus requisits de 5G QoS. Aquesta anàlisi es porta a terme mitjançant la caracterització de polítiques de control d'admissió amb un model de Markov i el modelat estadístic de l'assignació de recursos, ambdós suportant múltiples inquilins i múltiples serveis. En segon lloc, la tesi aborda el problema del control d'admissió de network slices proposant una metodologia per l'estimació dels recursos requerits per una RAN slice, que es basa en la anàlisi de dades. Aquesta metodologia dona suport a la decisió sobre l'admissió o rebuig de noves sol·licituds de creació de RAN slices. En tercer lloc, la tesi explora el potencial de la intel·ligència artificial, concretament, de les tècniques de Deep Reinforcement Learning (DRL) per a tractar el problema de la compartició de capacitat en escenaris amb RAN slicing. Amb aquest objectiu, es proposa i s'avalua una solució de compartició de capacitat basada en DRL que distribueix la capacitat disponible en un escenari multicel·lular entre múltiples inquilins. Aquesta solució es planteja com una solución de Multi-Agent Reinforcement Learning (MARL) basat en Deep Q-Network. Finalment, aquesta tesi tracta diversos aspectes relacionats amb la implementació de la solució de compartició de capacitat basada en DRL,

incloent-hi consideracions sobre la compatibilitat de la solució amb els estàndards, l'impacte de l'entrenament de la solució al seu comportament i rendiment, així com les eines i tecnologies necessàries per al desplegament de la solució en un entorn de xarxa real.

Acknowledgement

First and foremost, I would like to thank my thesis advisors, Dr. Oriol Sallent and Dr. Anna Umbert, for their guidance, support and dedication throughout this thesis. Their valuable and insightful comments have indubitably contributed to improve the quality of this dissertation. Besides my advisors, I would like to express my gratitude to Dr. Jordi Pérez-Romero, for his valuable help and involvement, and the rest of the members of the Mobile Communications Research Group (GRCM). Also, I cannot forget to thank my colleagues from GRCM for the good times we shared during coffee and lunch breaks. It has been a pleasure working with all of you in GRCM.

Furthermore, the work of this thesis has been supported by a FI AGAUR grant (Ref. 2018FI_B_00412 2019FI_B1_00102 and 2020FI_B2_00075) of the Secretariat for Universities and Research of the Ministry of Business and Knowledge of the Government of Catalonia and, during the last months, with the financial support of the GRCM to finish my dissertation, which I sincerely appreciate.

Last but not least, I would like to thank my family and friends, for supporting me from the very first moment I started this journey, giving me the strength I needed during tough times and celebrating my achievements as their own.

Contents

List of Figures

List of Tables

List of Abbreviations

3GPP	3rd Generation Partnership Project
5G	Fifth Generation
5GC	5G core
5QI	5G QoS Identifier
A3C	Asynchronous Advantage Actor Critic
AC	Admission Control
AI	Artificial Intelligence
AMF	Access and Mobility Management Function
ARP	Allocation and Retention Priority
C-RAN	Cloud-RAN
CDF	Cumulative Density Function
CN	Core Network
COTS	Commercial Off-The Shelf
CQI	Channel Quality Indicator
CTMC	Continuous Time Markov Chain
D2D	Device to Device
DDPG	Deep Deterministic Policy Gradients
DDQN	Double DQN
DNN	Deep Neural Networks
DPG	Deterministic Policy Gradients
DQN	Deep Q-Network
DRB	Data Radio Bearer
DRL	Deep Reinforcement Learning
eMBB	Enhanced Mobile BroadBand
FCAPS	Fault, Configuration, Accounting, Performance and Security
FH	Open Fronthaul
GAN	Generative Adversarial Network
GBR	Guaranteed Bit Rate
GFBR	Guaranteed Flow Bit Rate
HD	High Definition

ICIC	Inter-cell Interference Coordination
InH	Indoor Hotspot
InP	Infrastructure Providers
IOC	Information Object Classes
IoT	Internet of Things
ITU	International Telecommunication Union
K-NN	K-Nearest Neighbours
KPI	Key Performance Indicator
LCM	LifeCycle Management
LTE	Long Term Evolution
MAC	Medium Access Control
MARL	Multi-Agent Reinforcement Learning
MC	Mission Citical
MC PTT	Mission Critical Push to Talk
MDAF	Management and Data Analytics Function
MDAS	Management Data Analytics Services
MIMO	Multiple-Input Multiple-Output
ML	Machine Learning
mMTC	Massive Machine Type Communications
MnS	Management Services
MOI	Managed Object Instances
MSE	Mean Squared Error
MVNO	Mobile Virtual Network Operators
NAS	Non Access Stratum
NF	Network Functions
NFV	Network Function Virtualisation
NFVI	Network Function Virtualisation Infrastructure
NG-RAN	Next-Generation RAN
NR	New Radio
NRM	Network Resource Model
NSI	network slice instance
NSP	Network Slice Provider
NSSI	Network Slice Subnet Instance
NSSMF	Network Slice Subnet Management Function

O-CU-CP	O-RAN Central Unit – Control Plane
O-CU-UP	O-RAN Central Unit – User Plane
O-DU	O-RAN Distributed Unit
O-RU	O-RAN Radio Unit
OAM	Operations, Administration and Management
OFDMA	Orthogonal Frequency-Division Multiple Access
OS	Operation System
PDCP	Packet Data Convergence Protocol
PDF	Probability Density Function
PHY	Physical layer
PLMNid	Public Land Mobile Network Identifier
PM	Performance Management
PNF	Physical NF
PRB	Physical Resource Block
PS	Packet Scheduling
QoS	Quality of Service
RAB	Radio Access Bearers
RAN	Radio Access Network
RAT	Radio Access Technology
RB	Resource Block
RF	Radio Frequency
RLC	Radio Link Control
RL	Reinforcement Learning
RMa	Rural Macro-cell
RPC	Remote Procedure Call
RRC	Radio Resource Control
RRM	Radio Resource Management
RSI	RAN Slice Instance
RSMF	RAN Slicing Management Function
RT-RIC	Real-Time RAN Intelligent Controller
S-NSSAI	Single Network Slice Selection Assistance Information
SAC	Slice Admission Control
SD-RAN	Software Defined RAN
SDAP	Service Data Adaptation Protocol

SDN	Software-Defined Networking
SINR	Signal to Interference Ratio
SLA	Service Level Agreement
SMF	Session Management Function
SMO	Service Management and Orchestration
SON	Self-Organizing Network
SSBE	Steady-State Balance Equation
TD	Time Difference
TN	Transport Network
UE	User Equipment
UMa	Urban Macro-cell
UMi	Urban Micro-cell
UPF	User Plane Function
URLLC	Ultra-Reliable and Low-Latency Communications
VM	Virtual Machine
VNF	Virtual Network Function

Chapter 1. Introduction

1.1. Background

1.1.1. Fifth Generation Systems (5G)

The Fifth Generation (5G) of cellular networks was initially conceived to simultaneously support a wide range of application scenarios and vertical industries (e.g., automotive, transport, industry or health) [1]. The envisaged use cases for 5G, identified in the IMT-2020 vision by the International Telecommunication Union (ITU-R), were classified into three main 5G use cases [2]:

- **Enhanced mobile broadband (eMBB)**: Corresponds to the evolution of 4G mobile broadband services of today, which addresses the human-centric use cases for multi-media content, services and data. The demand for this kind of services will continue to increase and new application areas are emerging, establishing new and diverse requirements (e.g., higher data rates, hotspots, wide-area coverage, etc.).
- **Ultra-reliable and low-latency communications (URLLC):** Use cases with stringent requirements for latency, reliability and high availability. It includes both human (e.g., tactile internet, mission-critical applications) and machine-centric (e.g., vehicle-to-vehicle, industrial wireless control) communications.
- **Massive machine type communications (mMTC):** Characterised by a large number of connected devices with generally transmitting low volume and non-delayed sensitive data with low energy consumption, such as the required in sensor networks for smart grids or smart cities.

Fig. 1 includes the mapping of some examples of envisaged use cases to the main 5G use cases scenarios defined for the IMT-2020. Note that some use cases not exactly fit one of the main 5G use cases, such as the case of augmented reality, which requires the high data rates of eMBB and the low latencies of URRLC.

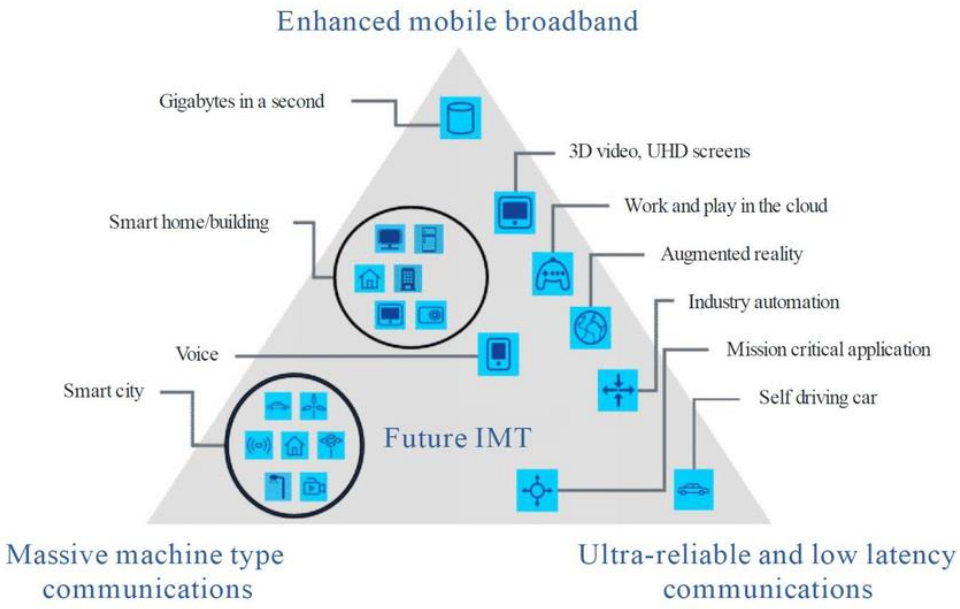


Fig. 1. IMT-2020 example use cases and main usage scenarios [2].

Despite this, the categorisation of the main use cases for 5G allowed identifying the main key capabilities required for 5G systems. The key capabilities and associated target values have been included in Fig. 2, which also depicts their level of relevance to each of the main use cases. In addition to the capabilities in Fig. 2, five more identified key capabilities are reliability, resilience, security and privacy, and operational lifetime. These capabilities are not included in the figure as they are not as easily quantifiable as the included in the figure, but are equally important. Notice that the challenging target values for the different key capabilities may not be simultaneously reached and some of them can be even mutually exclusive.

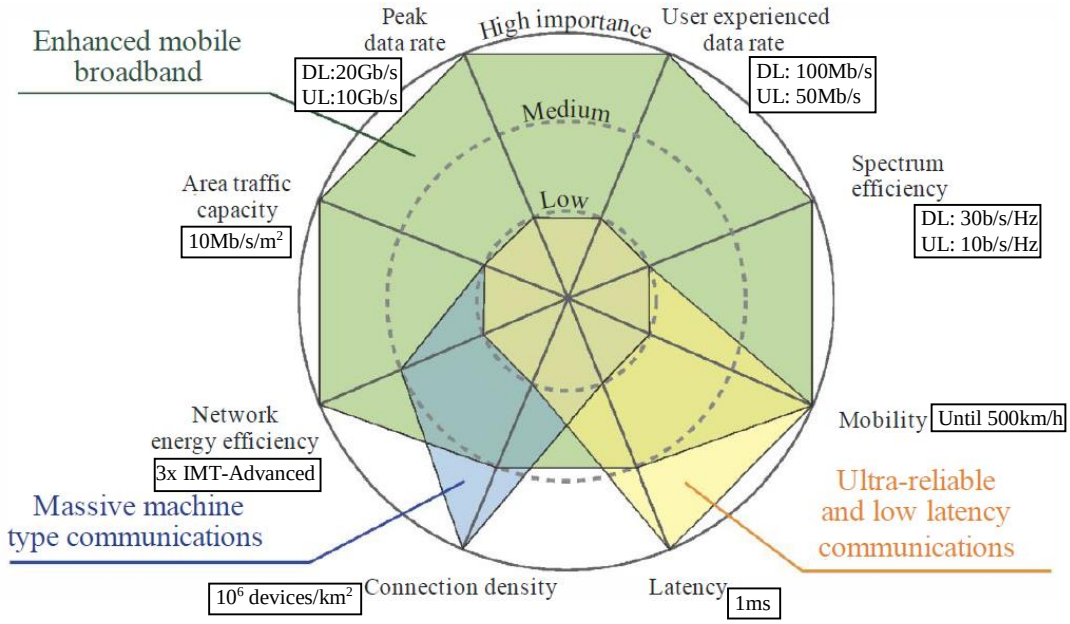


Fig. 2. Key capabilities, requirements and their importance to the different use scenarios [2].

Besides, 5G is envisaged to facilitate the evolution of current business models but also will enable new ones. Partnerships will be established on multiple layers ranging from sharing the infrastructure to exposing specific network capabilities as an end-to-end service and integrating partners' services into the 5G system

through a rich and software-oriented capability set [1]. The sharing of the mobile network infrastructure, owned by Infrastructure Providers (InPs), among multiple communication providers such as Mobile Virtual Network Operators (MVNOs) or vertical service providers, denoted as "tenants", is one of the main characteristics of 5G architectures since the sharing process will reduce capital and operational costs [3].

To enable the support of the identified use cases and associated requirements, as well as the envisaged business models and partnerships, the 3rd Generation Partnership Project (3GPP) initiated the development and standardisation of the new 5G systems architecture, defining a radio access network, known as Next-Generation RAN (NG-RAN) [4] with a new radio access technology, known as New Radio (NR), and, in parallel, the development of the new 5G core network, referred to 5GC [5]. NR incorporates new features and technologies to achieve the challenging target values, such as the use of millimetre-Wave (mm-Wave) frequencies, massive Multiple-Input Multiple-Output (MIMO) antennas or beamforming, among others [6]. Moreover, 5G architecture is defined as service-based architecture, focusing on network functionalities rather than network nodes, with decoupled control and user planes that supports network slicing, which is a central feature of 5G that is detailed in the following.

1.1.2. Network slicing

Previous mobile communication network architectures lack of the flexibility and scalability required to support the multi-tenancy and multi-service requirements of 5G. Therefore, network slicing is one of the key features of 5G architecture that relies on Software-Defined Networking (SDN) and Network Function Virtualisation (NFV) technologies [7], which combined enable the network softwarisation and provide the network with the required programmability, flexibility and modularity required for 5G. Based on this, network slicing allows creating end-to-end (self-contained) multiple logical networks, denoted as network slices, each tailored for a given use case, on the top of a common infrastructure [8]. Network slices are composed of mobile Network Functions (NF), as depicted in Fig. 3, which are appropriately combined to satisfy the requirements of a particular purpose or service category (e.g., applications with different access and/or functional requirements) or even individual customers (e.g., enterprises, third party service providers). Therefore, network slicing allows sharing a common network infrastructure among multiple tenants by providing each tenant with a network slice optimised to the tenants' specific requirements.

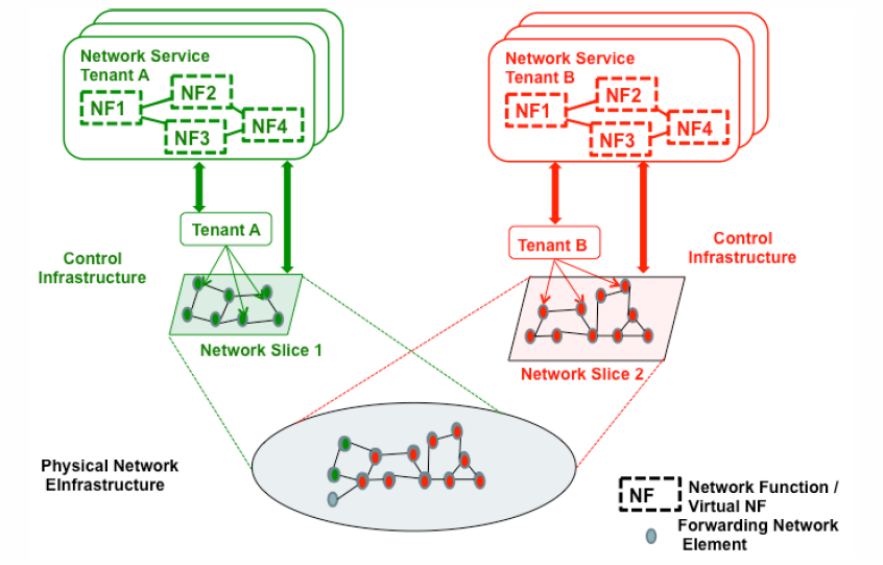


Fig. 3. Network slicing scheme [9].

Network slices are specified by a number of specific policies and configuration that allow supporting service-specific requirements such as the Radio Access Technology (RAT), bandwidth, latency, guaranteed Quality of Service (QoS), security level, etc. Moreover, a fundamental property of network slicing is the isolation, which assures the performance guarantees and security for each tenant when different tenants use network slices with conflicting performance requirements [10]. Each slice may be fully or partly isolated from other slices at different levels (physical, management, data), depending on the level of exposure requirements. Hence, the isolation can be achieved by using different physical resources, separating by virtualization means a shared resource and through sharing a resource with the guidance of a respective policy that defines the access rights for each tenant. In addition, network slicing can be composed of different hierarchy levels, enabling the possibility to create slices inside a higher-level slice, offering a greater abstraction with a broader scope. For example, a tenant who acquired a network slice from an InP, offers a partial amount of such resources to enable a utility provider that uses its virtual network to form an Internet of Things (IoT) slice.

Each network slice in the system is composed by a 5G Core Network (CN) part, denoted as CN slice, and a NG-RAN part, denoted as RAN slice [11]. The deployment of CN slices can be deployed by appropriately instantiating a combination of 5GC NFs for the different CN slices depending on the specific slice requirements [12]. These 5GC NFs may include different functionalities included in the 5GC architecture [5] for both control and user planes, such as the access and mobility management function (AMF), user plane function (UPF), the session management function (SMF). The 5GC NFs are expected to be implemented as virtualized network functions (VNFs) running on multi-tenant cloud infrastructures, which allow flexibly orchestrating the different CN slices and scaling the number of slices and combination of NFs for each of the CN slices.

Regarding RAN slices, their deployment needs to deal with the appropriate management of the common, and scarce, pool of radio resources (i.e., RF bandwidth) available at each of the NG-RAN nodes (i.e., gNB

or eNB) in order to simultaneously deliver multiple and diverse RAN behaviours required by the different RAN slices [13]. Therefore, the realization of RAN slices is an especially challenging task, which requires the definition of strategies and algorithms at diverse layers that allow managing the physical resources with the objective of maximizing the overall utility and the satisfaction of the requirements of the different RAN slices as is further discussed in the following section. In addition to the spectrum constraint, RAN slicing needs to address other aspects such as its operation when considering multi-RAT, the level of information exposure or sharing constraints when configuring and partitioning the slices. Also, the RAN programmability and virtualization are key elements to enable RAN slicing [10]. The RAN programmability and virtualization, also referred to as Software Defined RAN (SD-RAN), is a key attribute of RAN slicing abstracting the underlying RAN resources and facilitating open APIs toward third parties via a Service Orchestrator entity that dynamically manages the resources dedicated to a network slice. In turn, the RAN virtualization is based on the notion of base station softwarization, which allows certain RAN functions to run at remote cloud platforms and has gained momentum within the emergence of Cloud-RAN (C-RAN) concept.

1.1.3. RAN Slicing management.

One of the main challenges in implementing RAN slicing consists of designing and managing several slices, while guaranteeing the Service Level Agreement (SLA) for each of them. The implementation of RAN slices can be performed considering diverse levels of isolation, which depend on how the radio resources are shared and how the RAN slices are multiplexed [14]. Fig. 4 illustrates the trade-off between the level of isolation and sharing efficiency for different RAN slice multiplexing options. At the extreme left, a slice corresponds to a standalone network with its specific spectrum and infrastructure. In this case, the maximum level of isolation is achieved but at the cost of a low sharing efficiency with higher needs in terms of infrastructure deployment. At the extreme right, a slice that is only limited to the CN is depicted, where the RAN is slice unaware. In this case, high sharing is achieved but at the cost of low isolation where the actions for a RAN slice can have huge impact in the performance of the rest of slices. Intermediate solutions with slices going lower in the protocol stack when moving left are also possible. All these RAN slice multiplexing options are possible, and the adoption of one of them will depend on the requirements of the RAN slices.

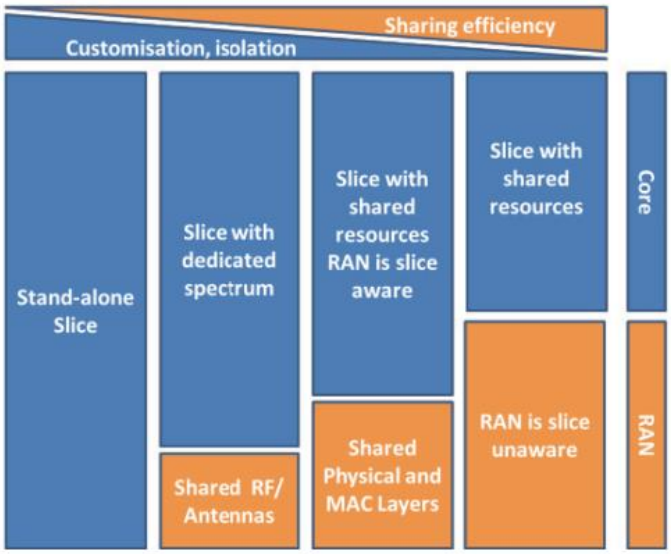


Fig. 4. RAN slice multiplexing options [14].

To enable the above RAN slice multiplexing options, the split of radio resources at RAN slice can be conducted by different Radio Resource Management (RRM) functionalities [15]:

- **Spectrum planning**: This function decides how the spectrum resources are arranged in carriers and how these carriers are assigned to the different cells. RAN slicing at spectrum planning can be implemented by arranging the spectrum resources into a number of carriers and assigning different carriers to each tenant, so that the required capacity and coverage by the tenant over the scenario is achieved. In this way, isolation is achieved at radio-electrical and traffic levels in the rest of RRM functionalities below.

- **Inter-cell Interference Coordination (ICIC):** This function intends to mitigate the inter-cell interference between neighbour cells that use the same carrier by determining the set of Resource Blocks (RBs), which constitute the basic physical radio resource unit for the dynamic allocation, that each cell is allowed to use in each of its allocated carriers as well as the maximum transmission power for each RB. RAN slicing at ICIC level can be implemented by establishing a tenant-specific ICIC policy to determine the set of RBs of each carrier that the tenant can use in the different neighbour cells to mitigate the inter-cell interference within the RAN slice of the tenant.

- **Packet Scheduling (PS):** This function decides how the set of available RBs for each carrier are used to transfer the data traffic of the established Radio Access Bearers (RABs), which are the data delivery services offered for data exchange between the user equipment (UE) and the mobile core network, and selects the physical layer parameters used in the RB transmission (e.g., modulation and coding scheme, antenna mapping in multi-antenna transmission, etc.), which controls the bit rate delivered to each RAB. At PS level, RAN slicing can be implemented by distributing the RBs among the tenants, so that traffic isolation is ensured within each cell. This admits the definition of tenant-specific PS algorithms

that allow deciding how the capacity provided to the slice is distributed among the RABs of the RAN slice.

- **Admission Control (AC):** This function determines whether the establishment request of a new RAB is accepted or rejected in a given cell, which is decided based on the QoS requirements of the active RABs and the requirements of the new RAB. At AC level, RAN slicing can be implemented by defining tenant-specific AC polices that decide whether a new RAB belonging to a tenant can be accepted or not. This decision considers the overall situation of all the tenants to ensure a proper split of the capacity.

The definition of algorithms for these RRM functionalities that consider the QoS requirements and SLA of RAN slices is identified as a challenge for the realization of RAN slices. Along with these RRM functionalities, new parameters and slice descriptors need to be introduced across the protocol layers of the NG-RAN nodes for the specification of the operation of the RRM functionalities, as discussed in [12].

Another relevant perspective for the realization of RAN slicing is from an orchestration and management view. 3GPP has already specified a management architecture for 5G systems that includes the operations and procedures for network and network slicing provisioning management services in [16], which includes the description of the LifeCycle Management (LCM) of network slices. For the case of network slices for NG-RAN (i.e., RAN slices) this embraces the creation, modification and termination of RAN slices, so that RAN slices can be dynamically managed over time through a network slicing management system. Given the complexity that implies the provisioning and management of RAN slices, Self-Organizing Network (SON) functionalities are identified as a key for the automation of the LCM of RAN slices. Remarkably, the automation of the LCM of RAN slices requires two main functionalities: slice admission control, which decides on the acceptance or rejection of a new RAN slice creation request with specific coverage, functional (i.e. features) and performance (e.g. service quality, capacity) requirements, and cross-slice resource optimization, which deals with the dynamic allocation of radio resources to RAN slices in each cell to ensure both the fulfilment of the RAN slice requirements and an efficient use of the available radio resources and can be used by RRM functionalities above.

1.2. Motivation, Objectives, Methodology and Contributions

Considering the above, the realization of network slicing in the RAN requires the definition of functionalities that allow the proper management of RAN slices and the configuration the NG-RAN nodes to support them. To this end, the adaptation of existing RRM functionalities at the different levels of the protocol stack to incorporate RAN slicing parameters has been identified as one of the potential solutions to support RAN slicing. Furthermore, the development of functionalities that enable the automatisation of the LCM of RAN slices, such as SON functions, is also envisaged to play a fundamental role to the realization of RAN slicing. To support these functionalities, Artificial Intelligence (AI) techniques are expected to be a key enabler due to its capability to deal with highly complex scenarios such as RAN slicing management. Nevertheless, the definition of RRM and AI-based solutions to support RAN slicing requires

the study of their requirements and limitations in order to be deployed in practice in the real network when supporting the envisaged multi-tenant and multi-service 5G scenarios. Also, the impact of these solutions on the achieved performance requires to be carefully assessed.

Therefore, **the main aim of this thesis is the definition and development of both RRM and AI-based solutions that allow evaluating the capacities of network slicing in the RAN and the potential of AI in the 5G scope**. To achieve this objective, the thesis has been divided in four main parts. The first part has focused on the assessment of RRM solutions at the different layers of the protocol stack. Specifically, an analytical model to assess admission control policies with RAN slicing capabilities at layer 3 and resource allocation solution at layer 2 has been proposed. This study has relied on the Markov characterization of admission control policies and a statistical modelling of the resource allocation for multi-tenant and multi-service scenarios, which has considered services with different 5G QoS requirements. The second part has addressed the problem of Slice Admission Control (SAC) by proposing a methodology for the estimation of the radio resources required by a RAN slice based on data analytics. Based on the estimated resources and considering the availability of radio resources, the decision on the admission or rejection of a RAN slice can be taken. The third part has focused on the problem of Cross Slice Resource Optimisation, referred hereto as the capacity sharing problem, which has been addressed by proposing a Multi-Agent Reinforcement Learning (MARL) capacity sharing solution based on Deep Q-Network (DQN), designed as a SON function. The solution aims at dynamically distributing of the capacity available in multi-cell scenarios among multiple tenants in order to fulfil the requirements of the different RAN slices and to efficiently use the resources available in the different cells. Based on this solution, insights into the impact and capabilities of DRL-based solutions for RAN slicing management have been analysed. In the fourth part, diverse implementation aspects regarding the application of DRL-based solutions for capacity sharing in RAN slicing scenarios have been discussed, including considerations on the compliance with O-RAN and 3GPP standards, the impact of the training on the performance, or the tools and technologies required for the deployment of these solutions in the real network environment.

The methodology followed during this thesis has included the definition of analytical models, their evaluation and validation through simulations, and their assessment in terms of the practicality.

With all the above considerations, the main thesis contributions (TC) can be summarized as follows:

[TC.1] Conception, development and evaluation of a Markov model for the characterization of the problem of RAN slicing for multi-tenant and multi-service scenarios. The developed Markov model allows the assessment of RRM policies with RAN slicing capabilities at different layers of the protocol stack and the relationships among the different dimensions of the RAN slicing problem, including aspects of the radio environment, services types and configurations, traffic scenarios, etc. The contributions of the Markov model are the following:

[TC.1.1] ***Definition of services in term of the 5G QoS of New Radio***. The Markov model includes the definition of services in terms of the Guaranteed Bit Rate (GBR), Allocation and Retention Priority (ARP) indicator and the 5G QoS Identifier (5QI) parameters. These QoS parameters play a fundamental role in the definition of the RRM functions included in the model.

[TC.1.2] ***Definition and assessment of Admission Control policies for RAN slicing.*** The Markov model allows studying the impact of AC policies at layer 3 in RAN slicing scenarios, where the admission control is performed at user level. AC policies with and without RAN slicing capabilities have been defined that allow achieving isolation between the admission of users belonging to different RAN slices.

[TC.1.3] ***Definition and assessment of a radio resource allocation model for RAN slicing.*** At layer 2, a statistical model that allows capturing 5G scenarios with variate radio propagation conditions has been included in the Markov model to characterize the radio resource allocation. The statistical model allows deriving the probability density function (pdf) of both the required resources and the assigned resources in multi-tenant and multi-service scenarios based on the QoS requirements.

[TC.1.4] ***Definition of Key Performance Indicators (KPIs) for the RRM policies assessment.*** A wide range of performance indicators have been defined that allow analysing the performance of the proposed RRM policies within the Markov model.

[TC.1.5] ***Validation of the model against a system-level simulator.*** The validity of the Markov model, which has been developed in MATLAB, has been assessed by using a system-level simulator that allows capturing multi-tenant and multi-service scenarios.

[TC.1.6] ***Evaluation of the RRM under different 5G relevant use cases and multiple cell deployment scenarios.*** The Markov model has allowed the assessment of the proposed RRM policies for eMBB and Mission Critical (MC) use cases under multiple cell deployments (e.g., urban microcell, rural macrocell), considering diverse QoS requirements of the services.

[TC.2] Conception, development and evaluation of a methodology to estimate the required radio resources by a RAN slice to support the SAC functionality. The proposed methodology supports the decision on the admission/rejection of slice admission requests when considering jointly the estimation of the required radio resources by a RAN slice and the available radio resources in a cell. The main contributions of the proposed methodology are the following:

[TC.2.1] ***Definition of a data analytics function to extract data information.*** The estimation procedure relies on the extraction of data analytics information from the cell performance measurements, which is fully aligned with the 3GPP vision for NG-RAN management.

[TC.2.2] ***Definition and assessment of the radio resource requirement estimation methodology for GBR and non-GBR services.*** The definition of the estimation methodology estimates the required radio resources by GBR and non-GBR services in order to satisfy the SLA of a RAN slice. This is performed by a statistical model that computes the pdf of the required resources by each service based on data analytics.

[TC.2.3] ***Evaluation of the estimation methodology considering diverse cell environments and services types.*** The estimation methodology has been assessed considering diverse service requirements under diverse cell environments, including 3GPP reference models and also representative cells and performance measurements extracted from a real network deployment.

[TC.3] Conception, development and evaluation of a Deep Reinforcement Learning-based Capacity sharing solution based on MARL and DQN for multi-tenant and multi-cell scenarios. The solution is able to dynamically provide the capacities to each tenant in the different cells in order to satisfy the fluctuating traffic demands of each tenant, satisfy the SLA established for each tenant and make an efficient use of the resources in each of the cells of the scenario. The main contributions of the proposed model are the following:

[TC.3.1] ***Definition a multi-agent scheme to deal with the capacity sharing problem with multiple tenants.*** In the proposed MARL approach, a DQN agent learns the policy for jointly assigning the capacities to be provided to a tenant in the different cells of the scenario. The different DQN-based agents operate in a synchronous and cooperative manner since their decisions are performed at the same time and each agent is designed to find a solution that benefits jointly all the tenants.

[TC.3.2] ***Definition of the DRL-based capacity sharing solution as a SON function.*** The practicality of the proposed approach is enforced with respect to previous capacity sharing solutions by formulating the solution as a SON function, which is integrated in a RAN slicing management framework well aligned with the on-going 3GPP standardisation work on management and orchestration of network slicing.

[TC.3.3] ***Validation of the DRL-based capacity sharing solution.*** The performance solution has been validated against reference solutions for capacity sharing in RAN slicing scenarios as well as against the optimum solution obtained through an exhaustive search algorithm.

[TC.3.4] ***Evaluation of the DRL-based capacity sharing solution under multiple scenarios.*** The solution has been assessed considering diverse cell deployments considering different requirements of the tenants. Also, diverse traffic distributions of the tenants among cells have been considered for the evaluation of the solution.

[TC.3.5] ***Assessment on the scalability of the solution.*** The evaluation of the MARL model has shown that the learning process can be conducted by a single agent and then the learnt policy

can be generalised and directly applied by the other agents, thus reducing the complexity of training in multi-agent scenarios and enhancing the scalability of the solution.

[TC.4] Assessment of implementation aspects of DRL-based capacity sharing solutions for RAN slicing management. A practical perspective on how to deploy the DRL-based capacity sharing solution for the RAN slicing management in the real network is provided, contributing as follows:

[TC.4.1] ***Compatibility analysis of RL-based solutions with the standards.*** Considerations on the definition of a compliant solution with the standardization held by 3GPP and O-RAN are provided and a compatible framework with these standards is proposed for the DRL-based capacity sharing solution.

[TC.4.2] ***Definition and evaluation of a methodology for analysing the impact of training on performance of the learnt polices by DRL-based capacity sharing solutions for RAN slicing.*** The defined methodology provides insights on the features and requirements of the data used for building training datasets and their impact on the achieved performance of the learnt policies during training and inference stages.

[TC.4.3] ***Design of the development and production stages of the DRL-based capacity sharing solution to be deployed in a real network environment.*** Insights into the software developed for the DRL-based capacity sharing solution based on Python and RL libraries are provided jointly with the design of a production solution to deploy the DRL-based capacity sharing solution in a real environment, which relies on containerization and the implementation of the required interfaces and protocols.

1.2.1. List of Publications

The above-listed contributions of the thesis have been published and/or submitted for publication in journals (J) and conferences (C). These dissemination activities and their relation to the above thesis' contributions are listed in the following:

1.2.1.1. Journals

[J1] **I. Vilà**, O. Sallent, A. Umbert, J. Pérez-Romero, "An Analytical Model for Multi-Tenant Radio Access Networks Supporting Guaranteed Bit Rate Services," in *IEEE Access*, vol. 7, pp. 57651-57662, April, 2019. Related to contribution [TC.1].

[J2] **I. Vilà**, J. Pérez-Romero, O. Sallent, A. Umbert, " Characterisation of Radio Access Network Slicing Scenarios with 5G QoS Provisioning," in *IEEE Access*, vol. 8, pp. 51414 - 51430, March, 2020. Related to contribution [TC.1].

[J3] R. Ferrús, J. Pérez-Romero, O. Sallent, **I. Vilà**, R. Agustí, "Machine learning-assisted cross-slice radio resource optimization: Implementation framework and algorithmic solution," in *ITU Journal on future*

and evolving technologies, vol. 1, nº1, pp.1-18, Dec. 2020. Related to contributions [TC.3] and [TC.4.1].

[J4] **I. Vilà**, J. Pérez-Romero, O. Sallent, A. Umbert, "A Multi-Agent Reinforcement Learning Approach for Capacity Sharing in Multi-tenant Scenarios," in *IEEE Transactions on Vehicular Technology,* vol. 70, no. 9, pp. 9450-9465, July 2021. Related to contribution [TC.3].

1.2.1.2. Conferences

[C1] **I. Vilà**, O. Sallent, A. Umbert and J. Pérez-Romero, "Guaranteed Bit Rate Traffic Prioritisation and Isolation in Multi-tenant Radio Access Networks," *2018 IEEE 23rd International Workshop on Computer Aided Modelling and Design of Communication Links and Networks* (CAMAD), Barcelona, 2018, pp. 1-6. Related to contribution [TC.1].

[C2] E. Jimeno, J. Pérez-Romero, **I. Vilà**, B. Blanco, A. Sanchoyerto and J. F. Hidalgo, "5G Framework for automated network adaptation in Mission Critical Services," *2018 IEEE Conference on Network Function Virtualization and Software Defined Networks* (NFV-SDN), Verona, Italy, 2018, pp. 1-5. Related to contribution [TC.1].

[C3] **I. Vilà**, J. Pérez-Romero, O. Sallent, A. Umbert and R. Ferrús, "Performance Measurements-Based Estimation of Radio Resource Requirements for Slice Admission Control," *2019 IEEE 90th Vehicular Technology Conference* (VTC2019-Fall), Honolulu, HI, USA, 2019, pp. 1-6. Related to contribution [TC.2].

[C4] J. Pérez-Romero, **I. Vilà**, O. Sallent, B. Blanco, A. Sanchoyerto, R. Solozábal, F. Liberal, "Supporting Mission Critical Services through Radio Access Network Slicing," in *6th International Conference on Information and Communication Technologies for Disaster Management* (ICT-DM'2019), Paris, France, Dec. 2019. Related to contribution [TC.2].

[C5] **I. Vilà**, J. Pérez-Romero, O. Sallent, A. Umbert, "A Novel Approach for Dynamic Capacity Sharing in Multitenant Scenarios," *31st International Symposium on Personal, Indoor and Mobile Radio Communications* (PIMRC2020-Fall), London, England (Online) Sept. 2020. Related to contribution [TC.3].

[C6] **I. Vilà**, J. Pérez-Romero, O. Sallent, A. Umbert, "A Deep Q-Network Approach for Radio Access Network Slicing," *XXXV Simposium Nacional de la Unión Científica Internacional de Radio* (URSI 2020), Málaga, Spain (Online). Sept. 2020. Related to contribution [TC.3].

[C7] **I. Vilà**, J. Pérez-Romero, O. Sallent, A. Umbert," Evaluation of a Multi-cell and Multi-tenant Capacity Sharing Solution under Heterogeneous Traffic Distributions," *2019 IEEE 93th Vehicular Technology Conference* (VTC2021-Spring*),* Helsinki, Finland (Online), April 2021. Related to contribution [TC.3].

[C8] **I. Vilà**, J. Pérez-Romero, O. Sallent, A. Umbert, "Impact Analysis of Training in Deep Reinforcement Learning-based Radio Access Network Slicing," 2022 *IEEE Consumer Communications and Networking Conference (CCNC 2022)*, Las Vegas (Online), January 2022. Related to contribution [TC.4.2].

1.2.2. Awards

The published articles in conferences have been recognised by the following awards:

- **Best Paper Runner Up Award** in recognition of the paper "Supporting Mission Critical Services through Radio Access Network Slicing" by 6th International Conference on Information and Communication Technologies for Disaster Management (ICT-DM'2019), Paris, December 2019.
- **Best Paper Runner Up Young Scientists Award** in recognition of the paper "A Deep Q-Network Approach for Radio Access Network Slicing" by XXXV Simposio Nacional de la Unión Científica Internacional de Radio (URSI 2020), Málaga, September 2020.

1.2.3. Relation to Research Projects

The work carried out during the thesis has been related to the following research projects funded by both Spanish and European Union entities:

- 5G-CLARITY: "Beyond 5G multi-tenant private networks integrating Cellular, WiFi, and LiFi, Powered by ARtificial Intelligence and Intent Based PolicY", funded by the Commission of European Communities. Ref. H2020-871428-5G-CLARITY.
- SONAR 5G: "Softwarization and automatic optimization of multi-tenant 5G radio access networks" funded by the Agencia Estatal de Investigación. Ref. TEC2017-82651-R.
- 5G ESSENCE: "Embedded Network Services for 5G Experiences" funded by the Commission of European Communities. Ref. H2020-761592-5G ESSENCE.

1.2.4. Other related activities

The work in this thesis has also been presented in the "Thesis in 4 minutes" competition held by Consell Interuniversitari de Catalunya of Generalitat de Catalunya. It has been selected as a candidate for the institutional final of Universitat Politècnica de Catalunya (UPC).

1.3. Thesis Outline

This thesis is organized in 6 chapters, whose structure is graphically summarized in Fig. 5.

Chapter 2 presents the Markov model characterisation of the RAN slicing problem in multi-cell and multi-service scenarios. After detailing the analytical definition of the Markov model, RRM policies with RAN slicing capabilities are defined at different layers of the protocol stack. At layer 3, an Admission Control policy that includes per-tenant capabilities is presented while a statistical model for modelling the resource allocation is formulated at layer 2. To assess the behaviour of these policies, several key performance metrics are formulated for the Markov model. The results extracted from the proposed solution are validated against those of a system level model. Furthermore, the proposed solution is evaluated considering different 5G use cases (eMBB and MC) and a wide range of scenarios in terms of the number of tenants and services, cell deployment, load conditions, etc.

Chapter 3 treats the SAC problem by proposing a methodology for estimating the required resources by a RAN slice based on data analytics. The problem of SAC is introduced and, accordingly, a functional framework for the analytics-based SAC functionality is proposed that is aligned with 3GPP work on network slices management. Afterwards, the different steps of the estimation methodology of the required radio resources by a RAN slice are described. In this regard, details on how data analytics are used to process the performance measurements extracted from cells are provided. Also, the resource estimation of the required resources by GBR and non-GBR services is formulated. The estimation methodology is evaluated in diverse cell environment, including both simulated-based cell environments and cells selected from a real LTE deployment. In addition, the potential of the methodology is highlighted by applying it on a relevant 5G use case scenario for supporting a Mission Critical Push to Talk (MC PTT) service during an emergence.

Chapter 4 deals with the problem of capacity sharing in RAN slicing scenarios by proposing a DRL-based solution, based on MARL and DQN. The problem of capacity sharing is presented and the DRL-based capacity sharing solution is formulated as a SON function. The validity of the solution is discussed by comparing the results to some reference solutions and the optimum solution obtained by an exhaustive search algorithm. Also, the proposed solution is evaluated considering single cell and multi-cell deployments and under diverse traffic demands conditions and distributions among cells.

Chapter 5 discusses on different aspects related to the implementation of DRL-based capacity sharing solutions for RAN slicing. First, the chapter highlights the relevance of defining DRL-based solutions that are compliant with the 3GPP and O-RAN standards and provides a plausible integration of the DRL-based capacity sharing solution proposed in Chapter 4 with these standards. Afterwards, an impact analysis of the training on the achieved performance of the learnt policies is conducted by defining a methodology to build training datasets with given features, whose performance is assessed for the proposed DRL-based capacity sharing solution based on defined performance metrics. Furthermore, insights into the development and production stages of the DRL-based capacity sharing solutions are provided, detailing the developed

software solution for the proposed DRL-based capacity sharing solution, as well as the proposed approach to deploy the solution in a real production environment.

Finally, Chapter 6 summarizes the work presented in this thesis and extracts the most important conclusions. Furthermore, future possible research directions are discussed related to the extension of the DRL-based capacity sharing and application of DRL solutions for the management of the RAN.

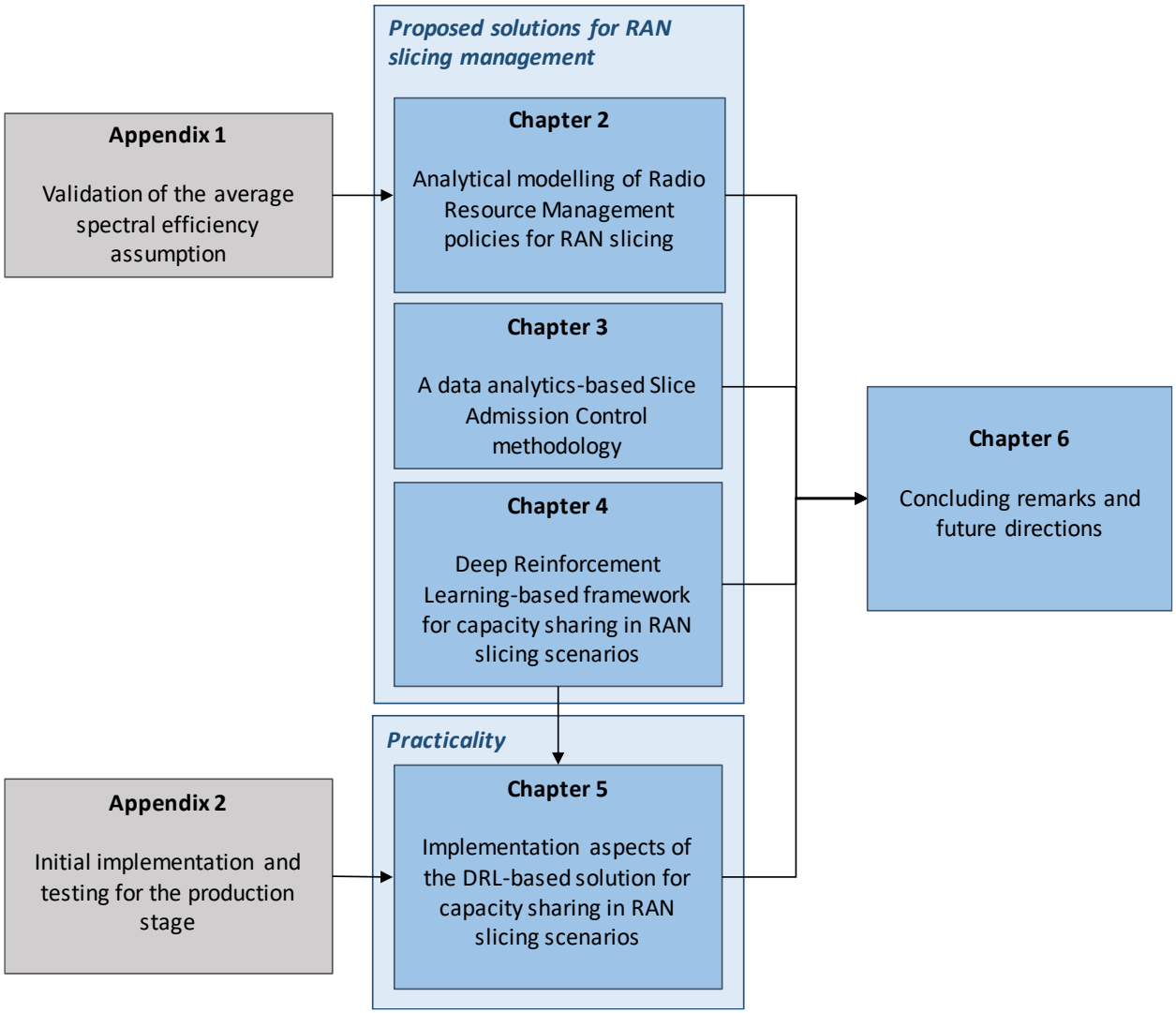


Fig. 5. Organisation of this Thesis.

Chapter 2. Analytical modelling of Radio Resource Management policies for RAN slicing

2.1. Introduction

This chapter addresses the RAN slicing problem from a modelling perspective by proposing and developing a Markov model characterisation of RAN slicing in multi-tenant and multi-service scenarios. The solution aims at assessing the performance of different Radio Resource Management (RRM) policies for RAN slicing at different layers of the protocol stack and establishing relationships among the different dimensions of the RAN slicing problem, including aspects of the radio environment, services types and configurations, traffic scenarios, etc. Specifically, the solution includes a Markov chain model to assess the behaviour of Admission Control (AC) policies at layer 3, where the admission control is performed at user level. Moreover, a resource allocation at layer 2 is also included, which relies on the statistical modelling of the required and assigned resources considering 5G scenarios with variate radio propagation conditions (i.e. diverse spectral efficiencies can be perceived by the different users in the system). The considered RRM policies have been properly characterised to provide slicing capabilities, so that isolation between slices is achieved both in the admission of users and the allocation of resources, and to support the definition of Guaranteed Bit Rate (GBR) and non-Guaranteed Bit Rate (non-GBR) services in terms of its corresponding 5G QoS parameters such as the Allocation and Retention Priority (ARP) and the 5G QoS Identifier (5QI) parameters. To assess the solution, key performance metrics of interest have been defined based on the analytical model. Based on these, the proposed model has been validated and evaluated under different RAN slicing scenarios, considering a wide variety of service-types and cell deployments.

The rest of the chapter is organized as follows. Section 2.2 provides the literature review of the RAN slicing problem from a RRM perspective. Afterwards, Section 2.3 describes the analytical model for RAN slicing scenarios by describing the considered system model in Section 2.3.1 and the Markov model approach in Section 2.3.2. The introduction of the considered admission control strategy and the resource allocation statistical model follow in sections 2.3.3 and 2.3.4, respectively. Then, Section 2.4 presents different performance metrics that can be extracted from the model, followed by a description of the implementation of the model in Section 2.5 and the validation of the model in Section 2.6. Section 2.7 includes the

evaluation of the model considering different scenarios and 5G use cases. Finally, Section 2.8 summarizes the main conclusions extracted from this chapter.

2.2. Literature review

Network slicing in the NG-RAN has been studied from multiple angles, ranging from virtualisation techniques and programmable platforms with slice-aware traffic differentiation and protection mechanisms [17][18][19] to algorithms for dynamic resource sharing across slices [20]. Similarly, [15] analyses the RAN slicing problem in a multi-cell network in relation to RRM functionalities and [21] proposes an adaptation algorithm for resource allocation, which is based on the deviations from requirements. In turn, [12] proposes a set of vendor-agnostic configuration descriptors intended to characterise the features, policies and resources to be put in place across the radio protocol layers of a NG-RAN node for the realisation of concurrent RAN slices. Also, [22] proposes a procedure to establish the level of centralisation of different RRM functions while [23] presents an adaptive inter-slice TDD allocation algorithm, where resources are assigned by minimising costs and interferences. Some other works focus on the network slice admission control for slices requests that need to support a given number of users for a certain time, such as [24][25], which target to optimise the infrastructure providers' revenue, or [26], which optimises the network utilisation by incorporating traffic forecasting capabilities. Moreover, frameworks for the realisation and study of network slicing have been implemented and tested such as in [27], which designs a system for the management of RAN slices and the provisioning of radio resources to slices based on their requirements in Long Term Evolution (LTE) technology, considering its extension to 5G NR.

In the above context, the RAN slicing problem is addressed here from a modelling perspective by proposing and developing a Markov model characterisation of RAN slicing in multi-tenant and multi-service scenarios. The Markov Chain theory is a statistical tool that allows studying the behaviour of systems before they are actually built by defining all the possible states in the system and their probabilistically based relations. By using this theory and appropriate numerical methods, it is possible to obtain relevant performance metrics of the system without the need of building it, allowing the prior refining of system aspects. Markovian approaches have been widely used to characterise the utilisation of resources in many fields, such as in mobility [28], cloud computing [29], Call Admission Control (CAC) scheme for 3G [30] or for heterogeneous networks RAT policies [31]. More recently, works in the field of 5G exploit Markov modelling to approach a proactive resource allocation scheme in highly mobile networks [32], the management of AC for handoff requests between small cell and macro cell domains [33], the computation of the estimated spectrum requirement [34] and the management of slices' creation [35]. Markov chain models have also been considered in [36] for spectrum sharing schemes and primary/secondary scenarios [37][38][39]. None of the above papers have considered the use of Markov chain models to study the performance of different RRM policies for RAN slicing at different layers of the protocol stack, which is treated in this chapter.

2.3. Analytical model for RAN slicing scenarios

2.3.1. System model

A scenario comprised of a common radio infrastructure shared by N tenants is assumed. Each tenant operates in a RAN slice, which has already been created and deployed by the Operations, Administration and Management (OAM) system among the RAN infrastructure by means of the Network Slice Subnet Management Function (NSSMF) [16]. The n-th tenant provides M_n service types, which can be either GBR or non-GBR services. The service type indicator $T_{s,n}$ takes value 0 if the s-th service of the n-th tenant is GBR and 1 if it is non-GBR. The QoS profile is given by the GBR value (i.e., the bit rate to be provided to the user of a GBR service, also referred to as Guaranteed Flow Bit Rate (GFBR) in 5G 3GPP's terminology), the ARP indicator [5], which defines the relative importance of the service requesting for resources and starts from 1 (highest priority) onwards (for successive lower priority services), and the priority level associated with the 5QI. Therefore, for GBR services, the QoS profile is characterised by the guaranteed $GBR_{s,n}$, the $ARP_{s,n}$ and the 5QI priority level $PL_{s,n}$ for s=1,…, M_n and n=1,…,N. In the case of non-GBR services, the $GBR_{s,n}$ is set to 0, as no data rate is guaranteed.

The considered scenario is depicted in Fig. 6. It is comprised of a gNB, which is the NG-RAN node operating the 5G NR interface, composed of a cell with a certain bandwidth subdivided in Physical Resource Blocks (PRB) of bandwidth B. Hence, the cell has a number of available PRBs N_{ava} at layer 1 to serve the User Equipment (UE) traffic demands.

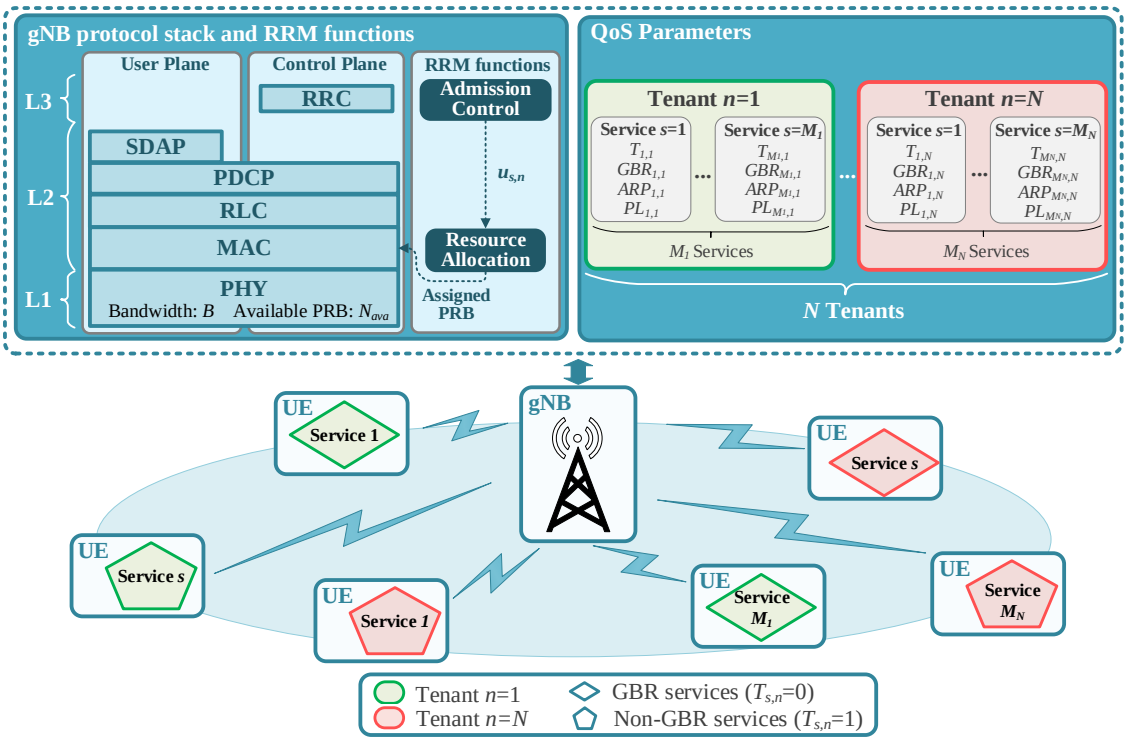


Fig. 6. System model conceptual scheme.

Fig. 6 also illustrates the different layers of the radio interface protocol stack at the gNB that determine how the user plane information and the control plane signalling is transferred between the UE and the gNB. Specifically, the transfer of the user plane information (e.g. IP packets associated to the services of the UE)

is carried out through the Service Data Adaptation Protocol (SDAP), Packet Data Convergence Protocol (PDCP), Radio Link Control (RLC), Medium Access Control (MAC) and physical (PHY) layers. In turn, the control plane signalling can be either generated at the Radio Resource Control (RRC) layer (e.g. for measurement reporting) or at upper layer Non Access Stratum (NAS) protocols for signalling between the UE and the 5GC network (e.g. for session establishment). In both cases, signalling is transferred between UE and gNB using the PDCP, RLC, MAC and PHY layers. Further details on the functionalities of each layer can be found in [6].

In order to efficiently use the radio resources and ensure the QoS requirements of the users in the system when transferring the information through the different layers of the protocol stack, the gNB includes a set of RRM functionalities, namely the AC function at Layer 3 (L3) and the resource allocation at Layer 2 (L2), which are briefly described in the following.

Whenever a new session of a service (i.e. a new QoS flow in 3GPP terminology) is established for transferring user data through the 5G system for a given UE, the 5GC network requests the gNB to set up resources to support this QoS flow at the radio interface. The 5GC network provides the gNB with the slice identifier of the tenant, i.e. the Single Network Slice Selection Assistance Information (S-NSSAI) [5], and the QoS parameters of the service. At the gNB, the SDAP layer maps the QoS flow to a Data Radio Bearer (DRB) that enables data transfer through the different layers of the radio interface protocol stack according to the expected QoS [4]. For this reason, and to make sure that the cell will have sufficient resources to support the requested QoS, the AC function at L3 is required to determine the acceptance or rejection of the new DRB in accordance with its QoS parameters and the available capacity.

Let us consider that each UE is provided with a single DRB and that, as a result of the AC function, the number of admitted DRB, and thus users, of the *s*-th service of the *n*-th tenant in the cell is $u_{s,n}$. Then, the resource allocation function is associated to the MAC layer at L2 and is in charge of dynamically assigning the N_{ava} available PRBs in the cell among the admitted DRBs, thus determining how the data of these DRBs travels through the physical layer.

In order to configure the multi-tenant behaviour of the AC and resource allocation functions, the OAM provides the RRM policy information to the gNB [16], including guidance for the split of radio resources among the different RAN slices. Specifically, let us assume that the OAM provides each gNB with the per-tenant parameters required to configure the RRM functionalities, as described in the detailed models for the L3 admission control and L2 resource allocation functions that are given in sections 2.2.3 and 2.2.4, respectively.

2.3.2. Markov model characterization

Assuming that users generate sessions of exponential duration according to a Poisson arrival process, the dynamic evolution of the number of admitted users of each service type and tenant can be characterised in general by a Continuous Time Markov Chain (CTMC) with $(M_1+M_2+\ldots+M_N)$-dimensional states. Let us

define $S_{(u_{1,1},\ldots,u_{M_1,1},u_{1,2},\ldots,u_{M_2,2},\ldots,u_{1,N},\ldots,u_{M_N,N})}$ as the state in which $u_{1,1},\ldots,u_{M_1,1},u_{1,2},\ldots,u_{M_2,2},\ldots,u_{1,N},\ldots,u_{M_N,N}$ users are admitted in the system. The state space is defined as:

$$S=\{S_{(u_{1,1},\ldots,u_{M_N,N})}\,|\,u_{s,n}\leq U_{\max,s,n}\} \tag{1}$$

where $U_{max,s,n}$ is the maximum allowed number of users of the *s*-th service of the *n*-th tenant, which is established for hardware limitation purposes (processor, memory, power). It is worth mentioning that the AC can further restrict the number of users per service to a value lower than $U_{max,s,n}$ depending on how the AC policy is specified.

Transitions between states occur due to session arrivals or session departures. In this respect, it is considered that session arrivals are generated with rate $\lambda_{s,n}$ for the *s*-th service of the *n*-th tenant, while the average session duration of this service is $1/\mu_{s,n}$. Moreover, since AC in L3 is in charge of admitting or rejecting users' requests depending on the system occupation, it also affects the transitions between states. In this respect, let us define $AC^{s,n}_{(u_{1,1},\ldots,u_{M_N,N})}$ as the binary AC indicator for the arrivals of the *s*-th service and *n*-th tenant, taking the value 1 if the new service request is accepted and 0 otherwise. It is worth mentioning that, since transitions can only occur due to the admission of a new session or the finalisation of an existing session, they can only increase or decrease the number of admitted users in one unit, meaning that transitions are only possible between neighbouring states. Therefore, the transition rate $q_{x,y}$ from a state x to another state $y\neq x$ is given by:

$$q_{x,y}=\begin{cases}\lambda_{s,n}AC^{s,n}_{(u_{1,1},\ldots,u_{M_N,N})} & \text{if } x=S_{(u_{1,1},\ldots,u_{s,n},\ldots,u_{M_N,N})} \text{ and } y=S_{(u_{1,1},\ldots,u_{s,n}+1,\ldots,u_{M_N,N})}\\ u_{s,n}\mu_{s,n} & \text{if } x=S_{(u_{1,1},\ldots,u_{s,n},\ldots,u_{M_N,N})} \text{ and } y=S_{(u_{1,1},\ldots,u_{s,n}-1,\ldots,u_{M_N,N})}\\ 0 & \text{otherwise}\end{cases} \tag{2}$$

where transitions to states with one more user (i.e. first condition in (2)) depend on the $AC^{s,n}_{(u_{1,1},\ldots,u_{M_N,N})}$ indicator and $\lambda_{s,n}$ while the transitions to a state with one less user (i.e. second condition in (2)) depend on the number of users of the service that decreases and $\mu_{s,n}$. The rest of transitions from/to states with more or less than one user are not allowed (i.e. third condition in (2)).

Fig. 7 illustrates a state transition diagram with all the possible transitions to/from a given state $S_{(u_{1,1},\ldots,u_{s,n},\ldots,u_{M_N,N})}$ when increasing or decreasing one user of each of the services. From the state transition diagram, the general Steady-State Balance Equation (SSBE) is given by:

$$\begin{aligned}&P_{(u_{1,1},\ldots,u_{s,n},\ldots,u_{M_N,N})}\Big[\sum_{s,n}u_{s,n}\mu_{s,n}+\sum_{s,n\ S_{(u_{1,1},\ldots,u_{s,n}+1,\ldots,u_{M_N,N})}\in S}\lambda_{s,n}AC^{s,n}_{(u_{1,1},\ldots,u_{s,n},\ldots,u_{M_N,N})}\Big]=\\ &\sum_{s,n}P_{(u_{1,1},\ldots,u_{s,n}-1,\ldots,u_{M_N,N})}\lambda_{s,n}AC^{s,n}_{(u_{1,1},\ldots,u_{s,n}-1,\ldots,u_{M_N,N})}+\sum_{s,n\ S_{(u_{1,1},\ldots,u_{s,n}+1,\ldots,u_{M_N,N})}\in S}P_{(u_{1,1},\ldots,u_{s,n}+1,\ldots,u_{M_N,N})}(u_{s,n}+1)\mu_{s,n}\end{aligned} \tag{3}$$

where $P_{(u_{1,1},\ldots,u_{s,n},\ldots,u_{M_N,N})}$ corresponds to the steady-state probability of being in state $S_{(u_{1,1},\ldots,u_{s,n},\ldots,u_{M_N,N})}$. When the SSBEs are obtained for all the states, the steady-state probabilities can be computed by using numerical methods capable of solving the system of equations composed by the different SSBEs and the following normalisation constraint:

$$\sum_{S_{(u_{1,1},\ldots,u_{M_N,N})}\in S} P_{(u_{1,1},\ldots,u_{M_N,N})} = 1 \tag{4}$$

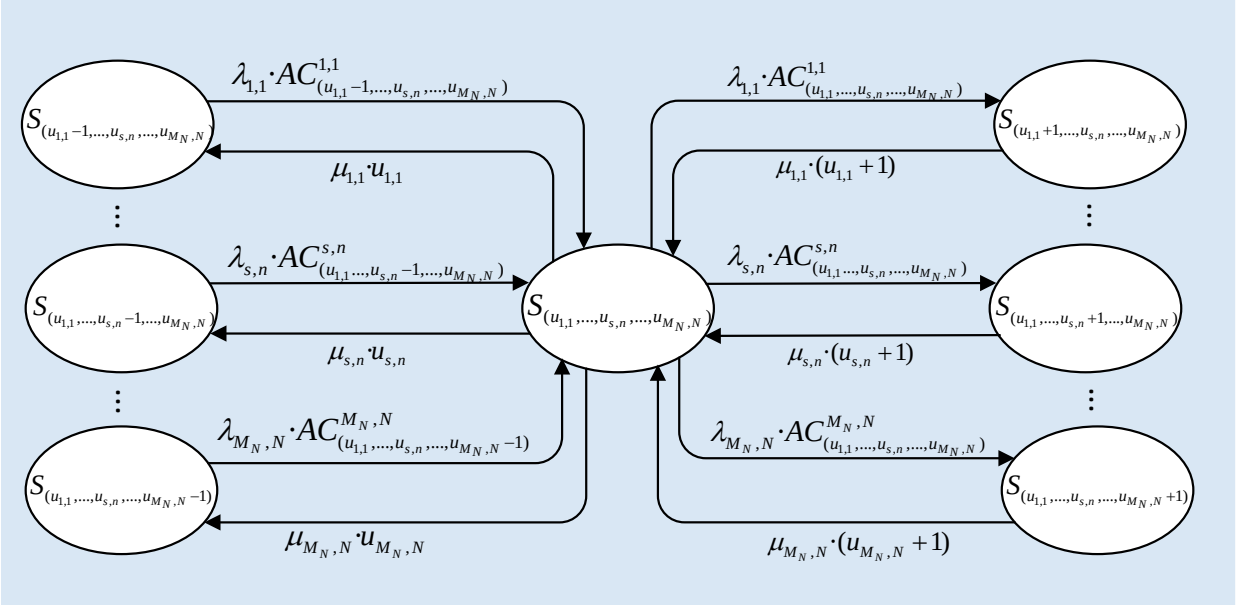


Fig. 7. State transition diagram with all the transitions from/to a given state $S_{(u_{1,1},\ldots,u_{s,n},\ldots,u_{M_N,N})}$ with $u_{1,1},\ldots,u_{s,n},\ldots,u_{M_N,N}$ users.

2.3.3. Admission control at Layer 3

The AC function at L3 decides on the acceptance or rejection of users initiating new sessions depending on their requested QoS and the already admitted users in the cell. Since the QoS requirements have different nature for GBR and non-GBR services, the considered AC policy behaves differently in each case.

For GBR services ($T_{s,n}$=0) the admission or rejection decision of a new user from the *s*-th service of the *n*-th tenant considers its requested $GBR_{s,n}$ and $ARP_{s,n}$ parameters and those of the already admitted GBR users of the same tenant, together with a per-tenant capacity threshold $C_{max,n}$ defined as the maximum aggregate GBR that can be admitted for tenant *n*. This threshold $C_{max,n}$ is provided by the OAM as the RRM policy information to be enforced by the AC. Correspondingly, the new GBR user can be admitted if the aggregate GBR considering both the new user and the already accepted users of tenant *n* with higher or equal priority than the new user (i.e. with ARP lower or equal than $ARP_{s,n}$) does not exceed the threshold $C_{max,n}$. It is worth noting that, in practice, $C_{max,n}$ could be dynamically adjusted, e.g. through a Self-Organizing Network (SON) function [40], to account for the spectral efficiency conditions experienced by the users in the cell.

In contrast, for non-GBR services ($T_{s,n}$=1) the system is not committed to guarantee any GBR value. Therefore, the AC in this case only checks that the number of admitted users does not exceed the maximum threshold $U_{max,s,n}$.

Based on these considerations, the AC decision for a new user of the *s*-th service of the *n*-th tenant is given by:

$$AC^{s,n}_{(u_{1,1},\ldots,u_{M_N,N})} = \begin{cases} 1 & \text{if } (T_{s,n}=0 \text{ and } \sum_{\substack{s'=1 \\ ARP_{s',n}\le ARP_{s,n} \\ T_{s',n}=0}}^{M_n} u_{s',n}\cdot GBR_{s',n} + GBR_{s,n} \le C_{\max,n}) \\ & \text{or } (T_{s,n}=1 \text{ and } (u_{s,n}+1)\le U_{\max,s,n}) \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

Note that the considered AC function is performed independently for each tenant, in the sense that the admission of a new user belonging to a certain tenant only depends on this tenant's parameters and occupation. Also, notice that the time scale at which AC is triggered depends on the session generation rate of the users $\lambda_{s,n}$, which will typically be in the order of seconds.

2.3.4. Radio resource allocation at Layer 2

From the perspective of the Markov model, admissions at L3 and session finalisations generate state transitions. In turn, within the sojourn time of a given state, the resource allocation at L2 defines how the N_{ava} available PRBs in the cell are assigned to the admitted users in a given state. In practice, this is performed by the Packet Scheduler (PS), which dictates in a short time basis (millisecond time scale) the user allocation of PRBs to transmit data packets by considering the QoS constraints and the actual radio conditions associated with each user. Here, the PS behaviour is characterised in accordance with the sojourn time of the Markov model states (i.e., typically few seconds) and, consequently, all the short time scale components of the PS (e.g. fast fading) will be averaged out in the resource allocation model.

Fig. 8 shows the considered resource allocation process. It is performed independently for each tenant *n* and assuming a maximum number of PRBs, $N_{th,n}$, to be allocated to this tenant, which is also determined by the OAM and provided to the gNB as RRM policy information. As depicted in Fig. 8, the process firstly allocates PRBs among the admitted users of GBR services. For this purpose, let us denote as $arp(1,n)$, $arp(2,n)$,...,$arp(M_n,n)$ the list of ARP values in increasing order for the GBR services of tenant *n*, i.e. starting by $arp(1,n)=min_{s|T_{s,n}=0}ARP_{s,n}$ and ending with $arp(M_n,n)=max_{s|T_{s,n}=0}ARP_{s,n}$. Then, the process firstly allocates the PRBs required by all the GBR services with $arp(1,n)$, then with $arp(2,n)$, and so on. As long as there are available PRBs to serve the users of an ARP value, resources are assigned. Instead, when not enough PRBs are available (i.e. there is congestion), the procedure proportionally distributes the available PRBs among the users of that ARP value. Other criteria might be adopted for handling persistent congestion situations such as the use of congestion control functions. However, they are out of the scope of this work. After this process, the remaining PRBs are allocated among the admitted users of non-GBR services. Both phases are further developed in the following.

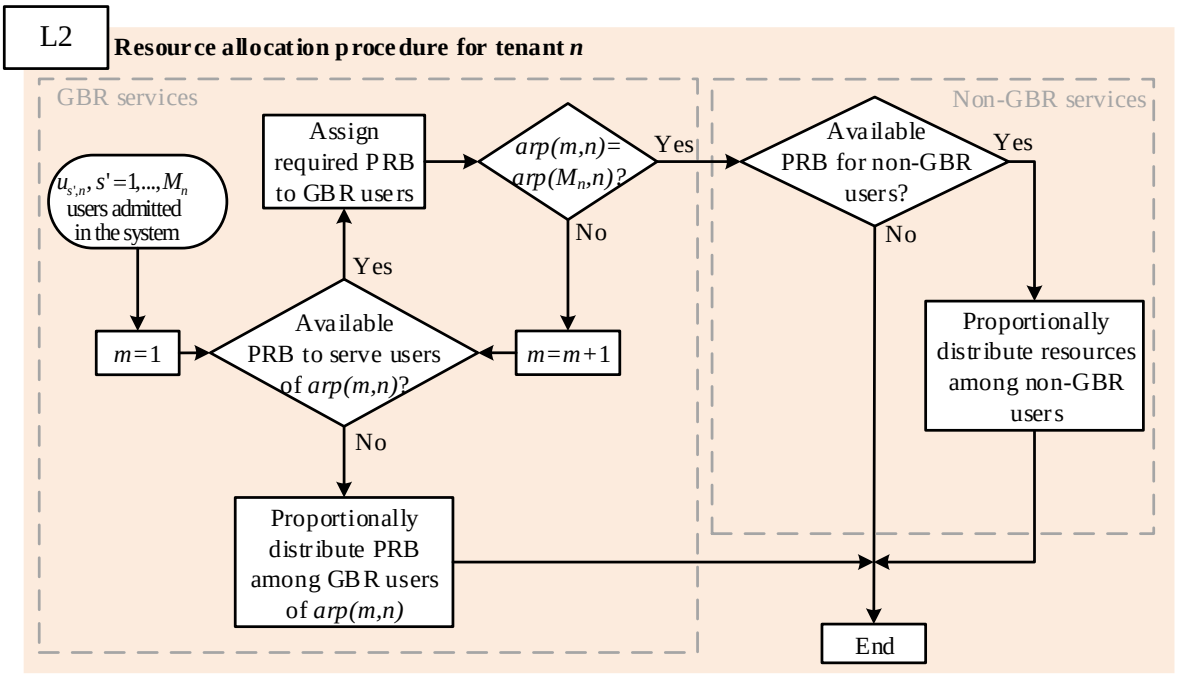


Fig. 8. Resource allocation procedure.

2.3.4.1. Resource allocation to GBR services

For a given state $x=S_{(u_{1,1},\ldots,u_{M_N,N})}$, the target here is to model the aggregate number of assigned PRBs to the users of a given GBR service s of tenant n, denoted as $a_{x,s,n}$. To obtain $a_{x,s,n}$, the first stage is to compute the number of required PRBs by the $u_{s,n}$ users. Then, the procedure of Fig. 8 is needed to allocate the $N_{th,n}$ available PRBs among the M_n service types, considering that the required PRBs of each service can be different and that the ARPs establish a prioritisation among services. The number of required PRBs by a user of the GBR service s of tenant n is given by:

$$N_{req,s,n} = \frac{GBR_{s,n}}{S_{eff} \cdot B} \tag{6}$$

where S_{eff} is the spectral efficiency, measured in b/s/Hz, and B is the PRB bandwidth. Since S_{eff} fluctuates depending on the propagation conditions that users experience when moving around the cell, it will be in general treated as a random variable. The resource allocation characterization has been conducted by considering two different assumptions regarding S_{eff}, firstly by considering S_{eff} as an average and, secondly, by considering the statistical distribution of S_{eff}, which are described in the following.

The first assumption considers S_{eff} as the average spectral efficiency that users would observe in the scenario, which allowed a low complexity formulation of the resource allocation. Under this assumption, $a_{x,s,n}$ is computed by iteratively performing the procedure in Fig. 8 for GBR users, starting by the users of lower ARP to the ones with higher ARP. As long as there are available resources to serve the users of a given ARP, each user gets the required resources $N_{req,s,n}$ and the available number of resources are reduced accordingly before moving to the next ARP. If ARP_{max} is reached, the spare resources remain unused. Instead, when there are not sufficient available resources to serve all the users of a given ARP (i.e., there is

congestion), the number of assigned resources $a_{x,s,n}$ to each user of this ARP is obtained by proportionally reducing $a_{x,s,n}$ according to the obtained ARP resource excess, given by:

$$a_{x,s,n} = N_{req,s,n}\left(1 - \frac{\left(\sum_{s',n' \mid ARP_{s',n'} \leq ARP_{s,n}} u_{s',n'} \cdot N_{req,s',n'}\right) - N_{th,n}}{\sum_{s',n' \mid ARP_{s',n'} = ARP_{s,n}} u_{s',n'} \cdot N_{req,s',n'}}\right) \quad (7)$$

The implications and accuracy of this first assumption are discussed in Appendix 1, which compares the results of the Markov model with a system level simulator that allows defining realistic scenarios where different scenarios can be configured in terms of cell deployment, propagation conditions and mobility. The comparison showed the validity of considering S_{eff} as an average holds for scenarios with low variation of S_{eff}. Instead, larger discrepancies are obtained between the results in the simulator and the Markov model for scenarios with large fluctuations on S_{eff}, indicating that the Markov model is not enough accurate for these cases.

Therefore, the second assumption considered the statistical distribution of S_{eff}. Specifically, based on measurements collected from the different users in the cell, it is possible to derive the pdf of the random variable $Y=1/(S_{eff} \cdot B)$, denoted as $f_Y(y)$. This pdf is obtained by gathering samples of the wideband Channel Quality Indicator (CQI) distribution [41]. The CQI is an integer index that indicates the modulation and coding scheme available to user in accordance to its experienced propagation conditions. The wideband CQI distribution is computed by the gNB from the CQI reports provided by the different users, and it is reported to a Management and Data Analytics Function (MDAF) [42] with a given periodicity (e.g. 15 minutes). The MDAF gathers samples of this distribution and averages them for a longer time period in order to get the adequate statistical validity to be representative of the cell conditions. Then, the averaged distribution of the CQI indices can be directly mapped to the distribution of the S_{eff} according to the tables in Section 5.2.2 of [43]. Specifically, the probability density function (pdf) of S_{eff}, denoted as $f_{Seff}(y)$, is obtained by associating the probability of a CQI index to the probability of the S_{eff} value corresponding to this CQI index. Based on $f_{Seff}(y)$, the pdf of Y, $f_Y(y)$ can be extracted based on $f_{Seff}(y)$ as:

$$f_Y(y) = f_{S_{eff}}\left(\frac{1}{y \cdot B}\right)\frac{1}{y^2 B} \quad (8)$$

Correspondingly, the number of required resources $N_{req,s,n}$ is another random variable, whose pdf is:

$$f_{N_{req,,s,n}}(k) = f_Y\left(\frac{k}{GBR_{s,n}}\right)\frac{1}{GBR_{s,n}} \quad (9)$$

Assuming that each user experiences independent propagation conditions, the pdf of the aggregate number of required PRBs $r_{x,s,n}$ of the s-th service of the n-th tenant in state x can be computed as:

$$f_{r_{x,s,n}}(r)=\left(\frac{1}{GBR_{s,n}}\right)^{u_{s,n}}\cdot \underbrace{f_Y\left(\frac{r}{GBR_{s,n}}\right)*\ldots* f_Y\left(\frac{r}{GBR_{s,n}}\right)}_{u_{s,n}} \tag{10}$$

where * represents the convolution operator.

As seen in Fig. 8, the allocation of resources to GBR users is performed in accordance with the priorities established by the ARP values of each service in the tenant, treating all the services with the same *arp(m,n)* together. Correspondingly, let us denote as $\mathbf{R_{arp(m,n)}}=[r_{x,s',n}|\ ARP_{s',n}=arp(m,n),\ T_{s',n}=0]$ the vector of required resources for the GBR services of tenant *n* with the same *arp*(*m*,*n*). Considering independence between requirements of different services, the joint pdf of the $r_{x,s,n}$ variables in $\mathbf{R_{arp(m,n)}}$ is:

$$f_{\mathbf{R_{arp(m,n)}}}\left(\mathbf{R_{arp(m,n)}}\right)=\prod_{\substack{s=1\\ ARP_{s,n}=arp(m,n)\\ T_{s,n}=0}}^{M_n} f_{r_{x,s,n}}(r_{x,s,n}) \tag{11}$$

In turn, let us denote as $Z_{arp(m,n)}$ the random variable representing the aggregated number of PRBs already assigned to the GBR services in tenant *n* of ARP lower than *arp*(*m*,*n*) and $f_{Z_{arp(m,n)}}(z)$ its pdf. For *arp*(1,*n*), since no resources have been yet assigned, $f_{Z_{arp(1,n)}}=\delta(z)$, where $\delta(\cdot)$ is the Dirac delta function.

Then, the pdf of the assigned resources $a_{x,s,n}$ of the *s*-th service of the *n*-th tenant with $ARP_{s,n}=arp(m,n)$, denoted as $f_{a_{x,s,n}}(k)$, can be computed by conditioning the value of assigned resources $a_{x,s,n}$ to the requirements $\mathbf{R_{arp(m,n)}}$ of the services with the same *arp*(*m*,*n*) and to the already assigned resources by GBR services $Z_{arp(m,n)}$. Then, by applying the law of the total probability, this yields to:

$$f_{a_{x,s,n}}(k)=\int_0^\infty\cdots\int_0^\infty f_{a_{x,s,n}|\mathbf{R_{arp(m,n)}},Z_{arp(m,n)}}(k\mid\mathbf{R_{arp(m,n)}},z)\cdot f_{Z_{arp(m,n)}}(z)\cdot f_{\mathbf{R_{arp(m,n)}}}(\mathbf{R_{arp(m,n)}})\cdot dz\cdot \prod_{\substack{s'=1\\ ARP_{s',n}=arp(m,n)\\ T_{s',n}=0}}^{M_n} dr_{x,s',n} \tag{12}$$

where $f_{a_{x,s,n}|\mathbf{R_{arp(m,n)}},Z_{arp(m,n)}}(k|\mathbf{R_{arp(m,n)}},z)$ is the pdf of $a_{x,s,n}$ conditioned to $\mathbf{R_{arp(m,n)}}$ and $Z_{arp(m,n)}$, which is given by:

$$f_{a_{x,s,n}|\mathbf{R_{arp(m,n)}},Z_{arp(m,n)}}(k\mid\mathbf{R_{arp(m,n)}},z)=\begin{cases}\delta(k-r_{x,s,n}) & \text{if } \sum_{\substack{s'=1\\ ARP_{s',n}=arp(m,n)\\ T_{s',n}=0}}^{M_n} r_{x,s',n}\le N_{th,n}-z\\ \delta(k-\alpha_{arp(m,n)}r_{x,s,n}) & \text{if } \sum_{\substack{s'=1\\ ARP_{s',n}=arp(m,n)\\ T_{s',n}=0}}^{M_n} r_{x,s',n}> N_{th,n}-z\end{cases} \tag{13}$$

The first condition in (13) considers the case that the number of available resources for the tenant is higher than the required resources given by $\mathbf{R_{arp(m,n)}}$ vector. Then, each user gets the required resources, i.e. the aggregate resources are $a_{x,s,n}=r_{x,s,n}$. In turn, the second condition in (13) considers the case that there are not sufficient resources for the tenant to fulfil the requirements of all the users of *arp(m,n)* (i.e. there is

congestion). In this case, the assigned resources are proportionally distributed among these users as $a_{x,s,n}=\alpha_{arp(m,n)}\cdot r_{x,s,n}$ with:

$$\alpha_{arp(m,n)}=\max\left(\frac{N_{th,n}-z}{\sum_{\substack{s=1\\ ARP_{s,n}=arp(m,n)\\ T_{s,n}=0}}^{M_n} r_{x,s,n}},0\right) \tag{14}$$

Once the computation of $f_{a_{x,s,n}}(k)$ for each of the services with *arp(m,n)* is completed, the pdfs for the subsequent ARP value *arp(m+1,n)* need to be derived. Then, the pdf $f_{Z_{arp(m+1,n)}}(z)$ is computed based on $f_{Z_{arp(m,n)}}(z)$ and the aggregate resources that have been assigned to the services with *arp(m,n)*, denoted by the random variable $A_{arp(m,n)}$. This yields to:

$$f_{Z_{arp(m+1,n)}}(z)=\int_0^{+\infty} f_{Z_{arp(m,n)}}(z')\cdot f_{A_{arp(m,n)}|Z_{arp(m,n)}}(z-z'|z')\cdot dz' \tag{15}$$

where $f_{A_{arp(m,n)}|Z_{arp(m,n)}}(t|z)$ is the pdf of $A_{arp(m,n)}$ conditioned to $Z_{arp(m,n)}$ given by:

$$f_{A_{arp(m,n)}|Z_{arp(m,n)}}(t|z)=f_{K_{arp(m,n)}}(t)\cdot H(t,N_{th,n}-z)+\delta(t-(N_{th,n}-z))\cdot\int_{N_{th,n}-z}^{\infty} f_{K_{arp(m,n)}}(k)dk \tag{16}$$

$$H(x,y)=\begin{cases}1 & x<y\\ 0 & x\geq y\end{cases} \tag{17}$$

where $f_{K_{arp(m,n)}}(z)$ is the pdf of the aggregate required resources for all the GBR services of *arp*(*m*,*n*), given by:

$$f_{K_{arp(m,n)}}(t)=\underbrace{f_{r_{x,1,1}}(t)*\ldots* f_{r_{x,s,n}}(t)}_{s=1,\ldots,M_n,\,ARP_{s,n}=arp(m,n),\,T_{s,n}=0} \tag{18}$$

Note that, for the case $m=M_n$, which corresponds to the maximum ARP value among the GBR services in tenant *n*, expression (15) gives the pdf of the random variable $Z_{GBR,n}$ that represents the aggregate PRBs assigned to all the GBR services in that tenant, i.e.

$$f_{Z_{GBR,n}}(z)=\int_0^{+\infty} f_{Z_{arp(M_n,n)}}(z')\cdot f_{A_{arp(M_n,n)}|Z_{arp(M_n,n)}}(z-z'|z')\cdot dz' \tag{19}$$

2.3.4.2. Resource allocation to non-GBR services

As seen in Fig. 8, the remaining PRBs after the allocation to GBR users of tenant *n* are distributed among the users of non-GBR services of the tenant. This distribution is carried out proportionally based on the 5QI priority level and the number of admitted users $u_{s,n}$ of each service, with the proportional constant $\sigma_{s,n}$, defined as:

$$\sigma_{s,n} = \frac{u_{s,n} \cdot (1/PL_{s,n})}{\sum_{\substack{s'=1 \\ T_{s',n}=1}}^{M_n} u_{s',n} \cdot (1/PL_{s'n})} \quad (20)$$

Consequently, the assigned PRBs to the non-GBR service s of tenant n are given by $a_{x,s,n}=(N_{th,n}-Z_{GBR,n})\cdot\sigma_{s,n}$. Therefore, the pdf of $a_{x,s,n}$ can be obtained by using the pdf of the assigned resources after GBR allocation $f_{Z_{GBR}}(z)$ as:

$$f_{a_{x,s,n}}(k) = \frac{1}{\sigma_{s,n}} f_{Z_{GBR,n}}(N_{th,n} - \frac{k}{\sigma_{s,n}}) \quad (21)$$

2.4. Performance metrics

Based on the steady-state probabilities, this section develops the different performance metrics of interest for the evaluation of the considered RRM strategies for layer 2 and 3.

2.4.1. Blocking probability

Blocking states are those in which the acceptance of a new user of a given service is not possible. Specifically, the set of blocking states for users of the s-th service of the n-th tenant is denoted as $S^b_{s,n}$, defined as:

$$S^b_{s,n} = \{S_{(u_{1,1},..,u_{M_N,N})} \in S \mid AC^{s,n}_{(u_{1,1},...,u_{M_N,N})} = 0\} \quad (22)$$

Similarly, the set of blocking states for the n-th tenant, S^b_n, are those states in which the acceptance of one user from any of the services of this tenant is not possible. Therefore, it is defined as the intersection of the sets of blocking states for the services of this tenant, i.e. $S^b_n = S^b_{1,n} \cap S^b_{2,n} \cap ... \cap S^b_{M_n,n}$. Similarly, the set of all blocking states in the system S^b is expressed as the intersection of the set of blocking states of each tenant/service.

Based on the blocking states, the blocking probability computed per service and per tenant is given by:

$$P^b_{s,n} = \sum_{S(u_{1,1},...,u_{M_N,N}) \in S^b_{s,n}} P_{(u_{1,1},...,u_{M_N,N})} \quad (23)$$

This can be easily extended to compute the blocking probability per tenant or the global blocking probability by considering S^b_n or S^b, respectively, in the summation of (23).

2.4.2. Occupancy metrics

Given the steady-state probabilities $P_{(u_{1,1},...,u_{M_N,N})}$, it is also possible to compute different metrics that provide information about the occupancy of the system. The average number of admitted users $\overline{U_{s,n}}$ of the s-th service of the n-th tenant is given by:

$$\overline{U_{s,n}} = \sum_{S_{(u_{1,1},\ldots,u_{M_N,N})} \in S} u_{s,n} \cdot P_{(u_{1,1},\ldots,u_{M_N,N})} \tag{24}$$

The average number of admitted users per tenant can be computed by adding the average number of users per service, i.e. $\overline{U_n}=\overline{U_{1,n}}+\overline{U_{2,n}}+\ldots+\overline{U_{M_n,n}}$. Similarly, the global system average number of admitted users $\overline{U}$ would be computed as the sum of the average number of users for all services and tenants.

Another system occupancy metric that can be obtained from the model is the average PRB utilisation $\overline{a_{s,n}}$ aggregated per each service. The average aggregated PRB utilisation in a given state $x=S_{(u_{1,1},\ldots,u_{M_N,N})}$ for the *s*-th service and the *n*-th tenant $\overline{a_{x,s,n}}$ is:

$$\overline{a_{x,s,n}} = \int_0^\infty k \cdot f_{a_{x,s,n}}(k) \cdot dk \tag{25}$$

Then, the system average aggregated PRB utilisation per service $\overline{a_{s,n}}$ can be computed by considering all the states $\overline{a_{x,s,n}}$ and the steady-state probabilities as:

$$\overline{a_{s,n}} = \sum_{S_{(u_{1,1},\ldots,u_{M_N,N})} \in S} \overline{a_{S_{(u_{1,1},\ldots,u_{M_N,N})},s,n}} \cdot P_{(u_{1,1},\ldots,u_{M_N,N})} \tag{26}$$

Accordingly, the average PRB utilisation per tenant results from $\overline{a_n}=\overline{a_{1,n}}+\overline{a_{2,n}}+\ldots+\overline{a_{M_n,n}}$ while the global system PRB utilisation $\overline{a}$ can be computed by adding the average PRB utilisation of each tenant. Then, the average normalised PRB utilisation of the *s*-th service of the *n*-th tenant $\overline{\omega_{s,n}}$ is expressed as:

$$\overline{\omega_{s,n}} = \frac{\overline{a_{s,n}}}{N_{ava}} \tag{27}$$

Similarly to the other metrics, the average normalised PRB utilisation per tenant $\overline{\omega_n}$ and the global average normalised PRB utilisation $\overline{\omega}$ result from adding the average normalised PRB utilisation of the tenant's services or all the services, respectively.

2.4.3. Average aggregated throughput

For a state $x=S_{(u_{1,1},\ldots,u_{M_N,N})}$, the average aggregated throughput for the *s*-th service of the *n*-th tenant $\overline{Th_{x,s,n}}$ is computed differently for GBR and non-GBR services. In the case of GBR services, $\overline{Th_{x,s,n}}$ is computed as:

$$\overline{Th_{x,s,n}} = \int_0^\infty \cdots \int_0^\infty k \cdot f_{Thr_{x,s,n}|\mathbf{R_{arp(m,n)}},Z_{arp(m,n)}}(k \mid \mathbf{R_{arp(m,n)}}, z) \cdot f_{Z_{arp(m,n)}}(z) \cdot f_{\mathbf{R_{arp(m,n)}}}(\mathbf{R_{arp(m,n)}}) \cdot dz \cdot \prod_{\substack{s'=1 \\ ARP_{s',n}=arp(m,n) \\ T_{s',n}=0}}^{M_n} dr_{x,s',n} \cdot dk \tag{28}$$

where $f_{Th_{x,s,n}|\mathbf{R_{arp(m,n)}},Z_{arp(m,n)}}(k|\mathbf{R_{arp(m,n)}},z)$ is the pdf of the aggregated throughput $\overline{Th_{x,s,n}}$ for the GBR service *s* from tenant *n* conditioned to $\mathbf{R_{arp(m,n)}}$ and $Z_{arp(m,n)}$, given by:

$$f_{Th_{x,s,n}|\mathbf{R_{arp(m,n)}},Z_{arp(m,n)}}(k \mid \mathbf{R_{arp(m,n)}}, z) = \begin{cases} \delta(k - u_{s,n} \cdot GBR_{s,n}) & \text{if } \sum_{\substack{s'=1 \\ ARP_{s',n}=arp(m,n) \\ T_{s',n}=0}}^{M_n} r_{x,s',n} \le N_{th,n} - z \\ \delta(k - \alpha_{arp(m,n)} \cdot u_{s,n} \cdot GBR_{s,n}) & \text{if } \sum_{\substack{s'=1 \\ ARP_{s',n}=arp(m,n) \\ T_{s',n}=0}}^{M_n} r_{x,s',n} > N_{th,n} - z \end{cases} \quad (29)$$

Expression (29) follows the same principle as (13), considering that, when the number of available PRBs is higher than the number of required PRBs (first condition in (29)), all users get the required PRBs and thus the throughput of each user is $GBR_{s,n}$. Instead, when there are not sufficient resources (second condition in (29)), the assigned PRBs are just a fraction $\alpha_{arp(m,n)}$ of the required ones and thus the throughput of each user is $\alpha_{arp(m,n)} \cdot GBR_{s,n}$.

Regarding non-GBR services and considering that the number of assigned PRBs to the non-GBR users depends on the spare PRBs after the allocation to GBR users, the average aggregated throughput $\overline{Th_{x,s,n}}$ for non-GBR users is independent of its actual spectral efficiency and is given by:

$$\overline{Th_{x,s,n}} = \overline{a_{x,s,n}} \cdot \overline{S_{eff}} \cdot B \quad (30)$$

where $\overline{S_{eff}}$ is the average of the spectral efficiency S_{eff}, derived from measurements in a similar way as variable Y.

The system average aggregated throughput $\overline{Th_{s,n}}$ for the s-th service of the n-th tenant can be computed by considering $\overline{Th_{x,s,n}}$ for all services and the steady-state probability as:

$$\overline{Th_{s,n}} = \sum_{S_{(u_{1,1},\ldots,u_{M_N,N})} \in S} \overline{Th_{S_{(u_{1,1},\ldots,u_{M_N,N})},s,n}} \cdot P_{(u_{1,1},\ldots,u_{M_N,N})} \quad (31)$$

The average aggregated throughput for the n-th tenant $\overline{Th_n}$ and the average global system aggregated throughput $\overline{Th}$ result from the summation of the average aggregated throughputs of the tenant's services or all the services in the system, respectively.

2.4.4. Degradation probability

Degradation occurs when congestion is reached and some GBR admitted users cannot be assigned with their required resources to provide $GBR_{s,n}$. Instead, they are assigned with a lower number of resources according to the considered resource allocation criteria.

For a given state x, the degradation probability of the GBR service s-th of the n-th tenant with $ARP_{s,n}$= $arp(m,n)$ can be obtained by the following expression:

$$p_{\deg,x,s,n} = \int_0^{\infty} \cdots \int_0^{\infty} \delta(k - \alpha_{arp(m,n)} r_{x,s,n}) \cdot H(N_{th,n} - z, \sum_{\substack{s'=1 \\ ARP_{s',n}=arp(m,n) \\ T_{s',n}=0}}^{M_n} r_{x,s',n}) \cdot f_{Z_{arp(m,n)}}(z) \cdot f_{\mathbf{R_{arp(m,n)}}}(\mathbf{R_{arp(m,n)}}) \cdot dz \cdot \prod_{\substack{s'=1 \\ ARP_{s',n}=arp(m,n) \\ T_{s',n}=0}}^{M_n} dr_{x,s',n} dk \quad (32)$$

where $H(\cdot)$ is defined in (17) and allows considering in the computation only those cases in which there are not enough resources to satisfy the service requirements (i.e. the second condition in (13)).

Based on the degradation at state level and the steady-state probabilities, the degradation probability of the *s*-th of the *n*-th tenant at system level is given by:

$$P_{s,n}^{\deg} = \sum_{x \in S} p_{\deg,x,s,n} \cdot P_{(u_{1,1},\ldots,u_{M_N,N})} \quad (33)$$

Note that the degradation probability is not computed for non-GBR services, as $GBR_{s,n}$ is not established for them.

2.5. Model implementation

Regarding the implemented method to compute the state probabilities and solving the SSBE, the model has been approached as a CTMC, so the following equation needs to be solved:

$$\underline{\boldsymbol{\pi}}^T \cdot \mathbf{Q} = \underline{\mathbf{0}} \quad (34)$$

where $\underline{\boldsymbol{\pi}}$ is a column vector containing all the steady-state probabilities $P_{(u_{1,1},\ldots,u_{M_N,N})}$, the superscript T denotes the transposed operation, and $\underline{\mathbf{0}}$ is a row vector with all elements equal to 0. The matrix $\mathbf{Q}$ is the state transition rate matrix, considering that rows define the origin state x and the columns the destination state y. The elements of $\mathbf{Q}$, referred to $q_{x,y}$, are defined by expression (2) for $y \neq x$, while for $y=x$ are given by $q_{x,y} = -\sum_{z \in S,\, z \neq x} q_{x,z}$.

The Gauss-Seidel method [44] has been selected to solve the SSBE system of equations (3) and compute the steady-state probabilities. This method avoids the discretisation of the CTMC transition rate matrix and provides a good compromise between accuracy and complexity in comparison to other methods. By employing the Gauss-Seidel method, expression (34) is transposed, leading to $\mathbf{Q^T} \cdot \underline{\boldsymbol{\pi}} = \underline{\mathbf{0}}^{\mathbf{T}}$ and then the matrix $\mathbf{Q^T}$ is decomposed as:

$$\mathbf{Q^T = D - (L + U)} \quad (35)$$

where $\mathbf{D}$ matrix is the diagonal of $\mathbf{Q^T}$ while $\mathbf{L}$ and $\mathbf{U}$ are, respectively, the strictly lower and upper triangular matrices of $\mathbf{Q^T}$. Then, the iterative matrix $\mathbf{H_{GS}}$ can be constructed according to:

$$\mathbf{H_{GS} = (D - L)^{-1} U} \quad (36)$$

Based on this matrix, the Gauss-Seidel method applies an iterative method that computes the state's probabilities in each iteration until reaching convergence. Specifically, the state probability at the *k*-th iteration, denoted as $\underline{\boldsymbol{\pi}}^{(k)}$, is computed as:

$$\underline{\boldsymbol{\pi}}^{(k+1)} = \mathbf{H}_{GS}\,\underline{\boldsymbol{\pi}}^{(k)} \tag{37}$$

where the initial probability vector $\underline{\boldsymbol{\pi}}^{(0)}$ is set randomly. Convergence is achieved at the first iteration *k* that fulfils the following condition based on the norm of successive iterates:

$$\left\| \underline{\boldsymbol{\pi}}^{(k)} - \underline{\boldsymbol{\pi}}^{(k-1)} \right\| < \varepsilon \tag{38}$$

where ε is the desired accuracy.

2.6. Model validation

The proposed model has been validated by comparing its results to the ones obtained with a custom made system-level simulator. The results from the Markov model have been extracted by developing it on MATLAB.

The system-level simulator used for the validation allows defining 5G multi-tenant and multi-service scenarios with diverse cell deployments and with different QoS requirements for each service. Sessions generation follow a Poisson distribution with exponentially distributed session duration. The implemented RRM functionalities in terms of AC and resource allocation in the simulator follow the same principles as in the Markov model, considering for the resource allocation the assumption that the spectral efficiency is statistically modelled as a random variable based on the measurements collected from the different users in the cell.

The adopted validation scenarios consider N=2 tenants, referred to *Tenant* 1 and *Tenant* 2, where *Tenant* 1 provides M_1=3 different services while *Tenant* 2 provides M_2=2 services. The QoS parameters of the different services are summarized in Table I, which have been specified according to the QoS model of [5]. The included parameters consist of the ARP, the PL and the GFBR, which specifies the GBR value to be provided to a QoS flow. The NG-RAN is composed by one gNB with a single cell. Two different radio environments are considered: a Urban Micro-cell (UMi) and a Rural Macro-cell (RMa) according to [45]. The configured parameters used to obtain the validation results for the UMi and RMa scenarios are summarized in Table II. Considering the defined 5G NR numerologies and channel bandwidth constraints for the different 5G NR bands [46], the largest channel bandwidth allowed to the considered RMa environment operating in the 700 MHz band is 20 MHz. Then, in order to ease the comparison of results in both RMa and UMi, the same 20 MHz bandwidth has been selected for both environments, which is composed by 51 PRBs [47], each one of B=360 kHz corresponding to 12 Orthogonal Frequency-Division Multiple Access (OFDMA) subcarriers with subcarrier separation of 30 kHz. Notice that, for both of the environments, the spectral efficiency samples of the different users are generated by mapping the Signal to

Interference Ratio (SINR) of users to spectral efficiency values according to the link-level model of section 5.2.7 of [48], which includes a parameter α to model modem implementation losses and link conditions such as the effects of error rates and retransmissions.

TABLE I. SERVICES PER TENANT FOR THE VALIDATION SCENARIO.

Tenant id (*n*)	Service id (*s*)	Type	GFBR	ARP	PL
1	1	GBR	5 Mb/s	1	30
	2	GBR	2 Mb/s	2	40
	3	Non-GBR	-	3	70
2	1	Non-GBR	-	3	80
	2	Non-GBR	-	3	60

TABLE II. MODEL CONFIGURATION PARAMETERS.

Environment	UMi	RMa
ISD (Inter-Site distance)	200 m	1735 m
gNB height	10 m	35 m
UE height	1.5 m	1.5 m
Minimum gNB-UE distance	10 m	35 m
Path Loss and Shadowing model	Model of sec. 7.4 of [45]	
Shadowing standard deviation in Line of Sight (LOS)	4	4
Shadowing standard deviation in Non-Line of Sight (NLOS)	7.82	8
Frequency	3.6 GHz	704 MHz
Total gNB transmitted power	37 dBm	45 dBm
gNB antenna Gain	Omnidirectional antenna with 5 dBi gain	
UE noise figure	9 dB	
Link-level model to map SINR and bit rate	Model in section. 5.2.7 of [48] with maximum spectral efficiency of 5.97 b/s/Hz (corresponding to SINR= 30 dB) and minimum SINR= -10 dB with α=0.6.	
Number of spectral efficiency samples for $f_Y(y)$ generation	10^7 samples obtained by simulating users at different random positions uniformly distributed within the cell and measuring their spectral efficiency.	
Average Spectral Efficiency	5.7 b/s/Hz	5.1 b/s/Hz
Cell available PRBs (N_{ava})	51 PRBs	
PRB Bandwidth (B)	360 kHz	
Cell maximum number of users	$U_{max,s,n}$=50 users for s,n=1,2	
Cell total capacity	104 Mb/s	94 Mb/s
L3 Maximum Capacity Threshold Tenant 1 ($C_{max,1}$)	62.4 Mb/s (corresponds to the 60% of the cell capacity)	56.4 Mb/s (corresponds to the 60% of the cell capacity)
L3 Maximum Capacity Threshold Tenant 2 ($C_{max,2}$)	N/A (only non-GBR services)	
L2 Maximum number of resources per tenant ($N_{th,1}$)	35.7 PRBs (corresponds to the 70% of the total cell PRBs)	
L2 Maximum number of resources per tenant ($N_{th,2}$)	15.3 PRBs (corresponds to the 30% of the total cell PRBs)	
Average session generation rate Tenant 1	varied from 0.001 to 0.1 sessions/s (corresponds to 0.27 Mb/s to 27.6 Mb/s of traffic offered load by GBR services)	
Average session generation rate Tenant 2	0.04 sessions/s	
Average session duration	120 s	
Session generation distribution Tenant 1	30% of session arrivals are of service 1, 40% of service 2 and 30% of service 3.	
Session generation distribution Tenant 2	40% of session arrivals are of service 1 and 60% of service 2.	

Table III contains the absolute percentage of error of the simulator results with respect to the Markov model results for all the services in the system for two session generation rates of Tenant 1. The small error percentages obtained show the suitability of the analytical model to evaluate RAN slicing scenarios with multiple tenants and services when considering diverse environments in terms of cell deployment (cell radius, transmitted power, etc.) and different traffic loads.

TABLE III. COMPARISON OF RESULTS BETWEEN MARKOV MODEL AND SIMULATOR.

Tenant id (n)	Service id (s)	Tenant 1 Session generation rates	UMi			RMa		
			% Error ($P^b_{s,n}$)	% Error ($\overline{a_{s,n}}$)	% Error ($\overline{Th_{s,n}}$)	% Error ($P^b_{s,n}$)	% Error ($\overline{a_{s,n}}$)	% Error ($\overline{Th_{s,n}}$)
1	1	0.02	0%	2.6%	3.3%	0%	1.9%	3.3%
		0.1	2.6%	0.6%	0.4%	2.0%	1.3%	0.5%
	2	0.02	0%	2.5%	1.5%	0%	2.2%	1.5%
		0.1	1.4%	1.7%	0.8%	0.2%	0.3%	0.7%
	3	0.02	-	0.0%	0.1%	-	0%	0.2%
		0.1	-	0.7%	0.6%	-	1.6%	1.8%
2	1	0.02	-	0.1%	0.0%	-	0.1%	0.1%
		0.1	-	0.2%	0.3%	-	0.7%	0.6%
	2	0.02	-	0.1%	0.1%	-	0.1%	0.2%
		0.1	-	0.0%	0.2%	-	0.4%	0.6%

2.7. Performance evaluation

This section provides the performance evaluation of the proposed model when considering different multi-tenant and multi-service scenarios under diverse cell environments. First, the proposed AC solution is compared to an AC benchmarking solution, which does not consider isolation among slices. Second, the model has been analyzed when considering a relevant use case envisaged for 5G, where GBR enhanced mobile broadband (eMBB) and GBR Mission Citical (MC) services provided by different tenants are analysed under standard and disaster traffic conditions. Third, the potential of the solution to deal with scenarios with different cell environments is assessed by evaluating the solution in a scenario with two tenants providing GBR and non-GBR services over urban micro cell and rural macro cell environments. Finally, the discussion of the impact of including a new tenant is provided.

2.7.1. Benchmarking of admission control strategy

The proposed AC solution in Section 2.3.3, which considers the per-tenant capacity threshold $C_{max,n}$ as well as the $GBR_{s,n}$ and $ARP_{s,n}$ parameters for the admission of a new GBR user, has been compared to a benchmarking AC solution, denoted as priority-based AC, which only considers the $GBR_{s,n}$ and $ARP_{s,n}$ parameters for the admission of a new user. Specifically, the priority-based AC considers a common admission control threshold C_{max} established for all the tenants, in addition to the $GBR_{s,n}$ and $ARP_{s,n}$ parameters, for the admission of a new GBR user from the s-th service of the n-th tenant, according to:

$$AC^{s,n}_{(u_{1,1},\ldots,u_{M_N,N})} = \begin{cases} 1 & \text{if } (T_{s,n}=0 \text{ and } \sum_{\substack{s'=M_n,n'=N \\ s',n'=1 \\ ARP_{s',n'} \le ARP_{s,n} \\ T_{s',n'}=0}} u_{s',n'} \cdot GBR_{s',n'} + GBR_{s,n} \le C_{\max}) \\ 0 & \text{otherwise} \end{cases} \quad (39)$$

The assumed scenario for the benchmarking analysis is comprised of a single cell that serves the users from N=2 Tenants providing M_n=2 GBR services each one. The configured parameters are summarised in Table

IV. For this analysis, the resource allocation considers the assumption that spectral efficiency is modelled as an average.

TABLE IV. MODEL CONFIGURATION PARAMETERS.

Parameter	Value	
Number of available PRBs (N_{ava})	25 PRB	
PRB Bandwidth (B)	180kHz	
Average Spectral Efficiency (S_{eff})	5.6 b/s/Hz.	
Data rate per PRB	1 Mbps/PRB	
Cell capacity	25 Mbps	
Guaranteed Bit Rate (GBR)	$GBR_{s,n}$=3Mbps for s,n=1,2	
Allocation and Retention Priority (ARP)	Configuration 1	$ARP_{1,1}$=1, $ARP_{2,1}$=2, $ARP_{1,2}$=3, $ARP_{2,2}$=4.
	Configuration 2	$ARP_{1,1}$=1, $ARP_{2,1}$=4, $ARP_{1,2}$=1, $ARP_{2,2}$=3.
	Configuration 3	$ARP_{1,1}$=1, $ARP_{2,1}$=3, $ARP_{1,2}$=2, $ARP_{2,2}$=3.
Average session generation rate	Tenant 1: varied from 0.001 to 0.06 sessions/s (corresponds to a variation from 0.36 Mb/s to 21.6 Mb/s) Tenant 2: - Low load: 0.02 session/s (corresponds to 7.2 Mbps) - High load: 0.035 sessions/s (corresponds to 12.6 Mb/s)	
Average session duration	120 s	
Tenant generation distribution	Tenant 1: 30% of generated traffic for service 1 and 70% for service 2. Tenant 2: 40% of generated traffic for service 1 and 60% for service 2	

2.7.1.1. Priority analysis

For evaluating the effect of the ARP value on the performance, the ARP configurations in Table II have been tested for the benchmarking priority-based AC policy with the relative capacity threshold ω_{max}=0.8, where ω_{max} is the ratio between C_{max} and the cell capacity. The offered load by Tenant 2 is set to the Low load level of Table I while Tenant 1's load is varied to observe different system load situations.

Fig. 9 shows the results obtained for the different services in the system considering the proposed ARP configuration in terms of blocking and degradation probability, both expressed in %. The comparison of the blocking probability of the different services reveals that higher blocking percentages are reached for those services with higher ARP value. This is the case of service 2 from Tenant 1 in configuration 1 or service 2 from Tenant 2 in configuration 2. In configuration 3, as services 2 from both Tenants share the same ARP value, which is the highest one in the system, the same performance in terms of blocking probability is found.

Focusing on the degradation probability, it is observed that the reached values are quite low (i.e. less than 3.5%) even for the highest considered load. This means that the guaranteed GBR of the admitted users is satisfactorily preserved. However, when the load is high, some differences can be perceived depending on the priority assigned to each of the services. Services with low ARP values are slightly degraded while services with high ARP values suffer from higher degradation. In addition, the effect of different loads can be noticed from Fig. 9f, where service 2 from Tenant 1 suffers from higher degradation than service 2 from Tenant 2, although both services have the same ARP value. The reason of this is that service 2 from Tenant

1 is more demanding because, as seen in Table I, 70% of the traffic from Tenant 1 belongs to service 2 while the traffic belonging to service 2 from Tenant 2 represents only 60%.

Based on these results, it can be concluded that the priority-based AC policy is capable of providing the required GBR to the admitted users with reduced degradation while differentiating between each service priority according to its ARP.

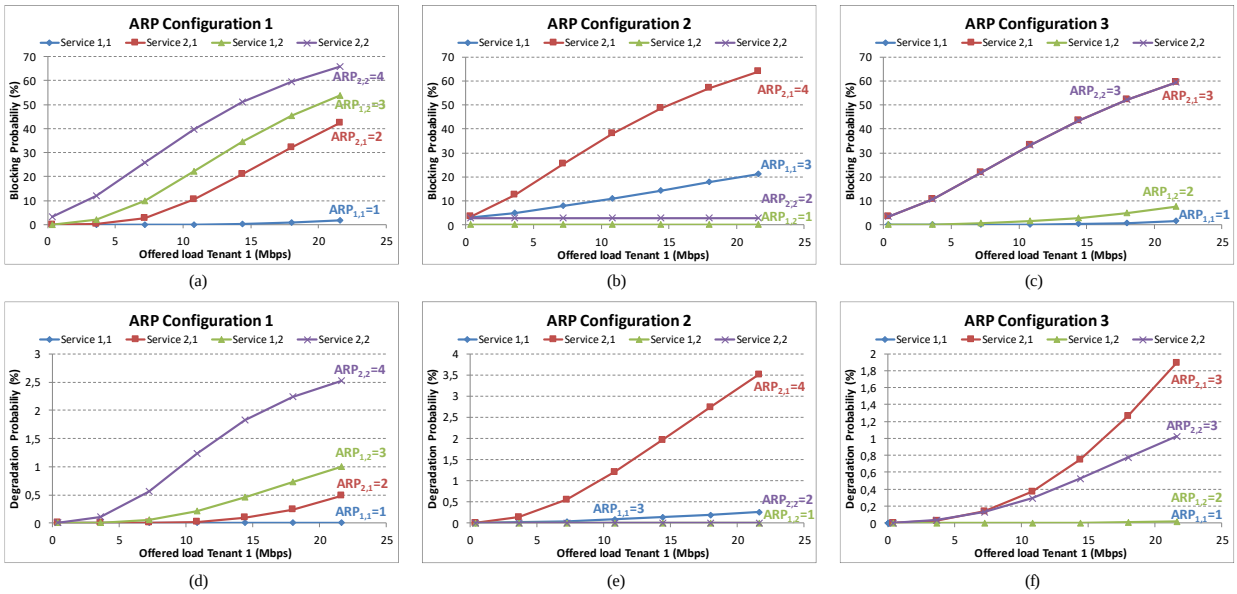


Fig. 9. Blocking probabilities for the different ARP configurations (a), (b), (c) and degradation probabilities for each configuration (d), (e), (f).

2.7.1.2. Priority and isolation analysis

The impact of proposed AC policy in Section 2.3.3 on the achieved performance has been analysed by comparing different load conditions for two different AC per-tenant threshold configurations: configuration 1 sets $\omega_{max,1}$=0.4 for Tenant 1 and $\omega_{max,2}$=0.4 for Tenant 2 while configuration 2 sets $\omega_{max,1}$=0.6 and $\omega_{max,2}$=0.2, where $\omega_{max,n}$ is the ratio between $C_{max,n}$ and the cell capacity. The selected ARP configuration corresponds to configuration 3 in Table IV and both the Low and High load levels of Tenant 2 have been studied.

By comparing the results obtained for Tenant 1 and 2, it is clearly seen that, with this AC policy, and as a difference from the benchmark priority-based AC policy, the load variation of Tenant 1 has no impact on the blocking probability of Tenant 2, which remains constant for all the offered loads of Tenant 1 (see Fig. 10c, Fig. 10d). Moreover, no variation is found in the performance of Tenant 1 (see Fig. 10a, Fig. 10b) for the Low and High load values of Tenant 2. These results reflect the isolation achieved between tenants when the proposed AC policy is applied.

Focusing on the effect of the per-tenant AC threshold variation, it is observed that blocking probabilities of Tenant 1 are higher when the threshold configuration 1 is used, as the AC threshold $\omega_{max,1}$ is lower, so less Tenant 1 users can be admitted to the system. On the contrary, blocking probabilities for Tenant 2 are lower for the threshold configuration 1, since the value of $\omega_{max,2}$ is higher for this configuration.

As expected, the effect of ARP values is also observed in the provided results, as service 2 from both Tenants, which has the higher ARP value, also perceives higher blocking probabilities than service 1. When comparing the results for Low Tenant 2 load and configuration 1 in Fig. 10 and the results in Fig. 9c, it can be noticed that the blocking probabilities of Tenant 1 for the priority and isolation-based AC policy are in general higher than for the priority-based AC policy. This is because in the later approach, Tenant 1 improves its performance at the expense of Tenant 2 since no distinction in the resources used by each tenant is performed in the AC, while in the priority and isolation-based AC, admissions of Tenant 1 only consider the threshold defined for that tenant. Therefore, performance is dependent on how this threshold is configured. For example, with configuration 2 (see Fig. 10a, Fig. 10b), performance is improved as $\omega_{max,1}$ better fits with the actual Tenant 1 load.

According to this, AC thresholds need to be configured according to the expected load for the tenant. The above observations reflect that the proposed AC policy in Section 2.3.3 effectively avoids that the overload in one Tenant may affect the other tenant while still respecting the priority established to each of the services of a Tenant. However, AC thresholds need to be properly defined in relation to the tenant traffic load.

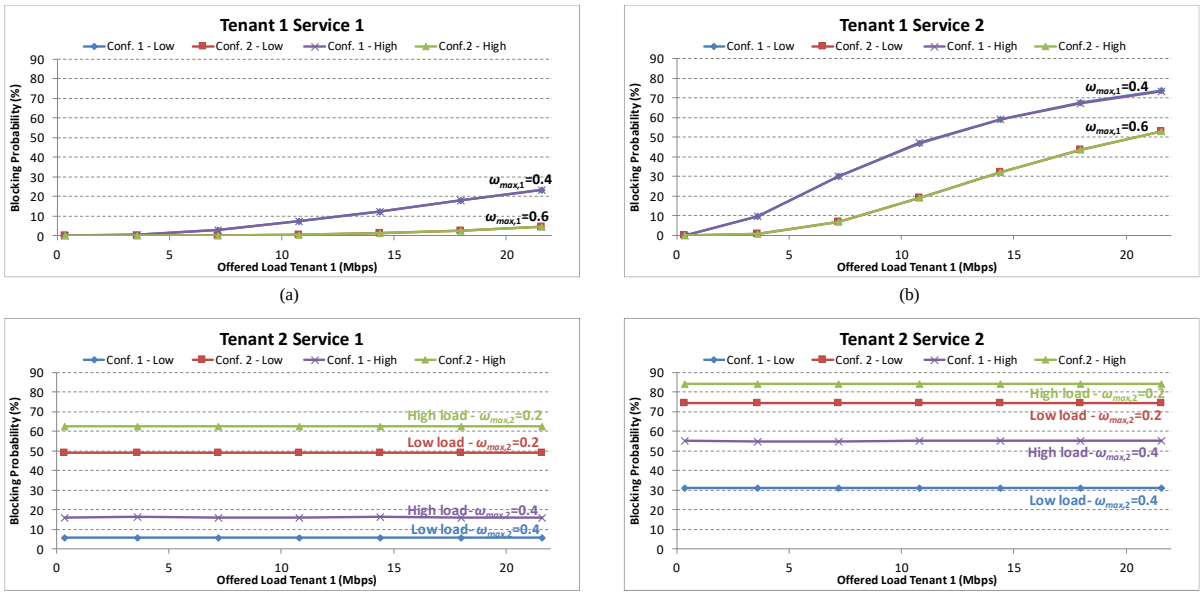


Fig. 10. Blocking probabilities for (a) service 1 and (b) service from Tenant 1 and (c) service 1 and (d) service 2 from Tenant 2 for configurations 1 and 2.

2.7.2. Assessment of the solution for GBR eMBB and MC services

The Markov solution has also been analysed in a relevant 5G use case scenario that considers a commercial operator that has deployed a NG-RAN in order to provide eMBB services to its users. Meanwhile, the commercial operator leases its infrastructure to a public safety operator serving MC communications. Therefore, the NG-RAN serves two different tenants, referring the commercial operator as *Tenant* 1 and the public safety operator as *Tenant* 2. In addition, each operator provides two different GBR services: Tenant 1 includes two video profiles, a basic profile with a standard quality and a premium profile with High Definition (HD) quality, whereas Tenant 2 provides two MC services, namely, MC Video and MC Push to Talk (MC PTT).

For each of the services, the QoS parameters summarised in Table V have been specified according to the QoS model of [5]. The included parameters consist of the ARP and the GFBR.

TABLE V. SERVICES PER TENANT.

Tenant	Tenant id (n)	Service	Service id (s)	Type	ARP	GFBR
Commercial operator	1	Premium – Video HD	1	GBR	2	10 Mb/s
		Basic - Video	2	GBR	3	3 Mb/s
Public safety operator	2	MC Video	1	GBR	2	5 Mb/s
		MC PTT	2	GBR	1	48 kb/s

The NG-RAN is composed by one gNB, with a single cell operating in a 20 MHz channel composed by 51 PRBs [46], each one of B=360 kHz corresponding to 12 OFDMA subcarriers with subcarrier separation of 30 kHz, which is one of the numerologies defined for 5G NR [47]. The configured parameters used to obtain the different results included in this section are summarised in Table VI. For the resource allocation, the spectral efficiency has been considered as an average.

TABLE VI. MODEL CONFIGURATION PARAMETERS.

Parameter	Value	
Number of available PRBs (N_{ava})	51 PRB	
PRB Bandwidth (B)	360 kHz	
Spectral Efficiency (S_{eff})	8.5 b/s/Hz	
Data rate per PRB	3 Mb/s/PRB	
Cell Maximum number of users	$U_{max,s,n}$=50 users for s,n=1,2	
Cell total capacity	156 Mb/s	
Scenario 1: Standard		
Average session generation rate	Tenant 1	Varied from 0.001 to 0.12 sessions/s (corresponds to a variation of 0.6Mb/s to 82.8 Mb/s)
	Tenant 2	0.05 session/s (corresponds to 6.2 Mb/s)
Average session duration	120 s	
Maximum capacity threshold	Tenant 1	$C_{max,1}$=93.6Mb/s (corresponds to the 60% of the total capacity)
	Tenant 2	$C_{max,2}$=31.2 Mb/s (corresponds to the 20% of the total capacity)
Tenant generation distribution	Tenant 1	40% of session arrivals are of service 1 and 60% of service 2
	Tenant 2	20% of session arrivals are of service 1 and 80% of service 2
Scenario 2: Disaster		
Average session generation rate	Tenant 1	0.04 session/s (corresponds to 17.76 Mb/s)
	Tenant 2	Varied from 0.001 to 0.25 sessions/s (corresponds to a variation of 0.36 Mb/s to 88 Mb/s)
Average session duration	120 s	
Maximum capacity threshold	Tenant 1	$C_{max,1}$=46.8 Mb/s (corresponds to the 30% of the total capacity)
	Tenant 2	$C_{max,2}$=78 Mb/s (corresponds to the 50% of the total capacity)
Tenant generation distribution	Tenant 1	10% of session arrivals are of service 1 and 90% of service 2
	Tenant 2	60% of session arrivals are of service 1 and 40% of service 2

Two different scenarios are defined in terms of load: a standard scenario, in which the traffic coming from the public safety operator is low while the traffic from the commercial operator is progressively increased, and a scenario representing a situation when an emergency or disaster has occurred and therefore the public safety traffic generated is varied until reaching high values whereas the traffic from the commercial operator remains moderate. Different maximum tenant capacity thresholds $C_{max,n}$ are configured: for the standard

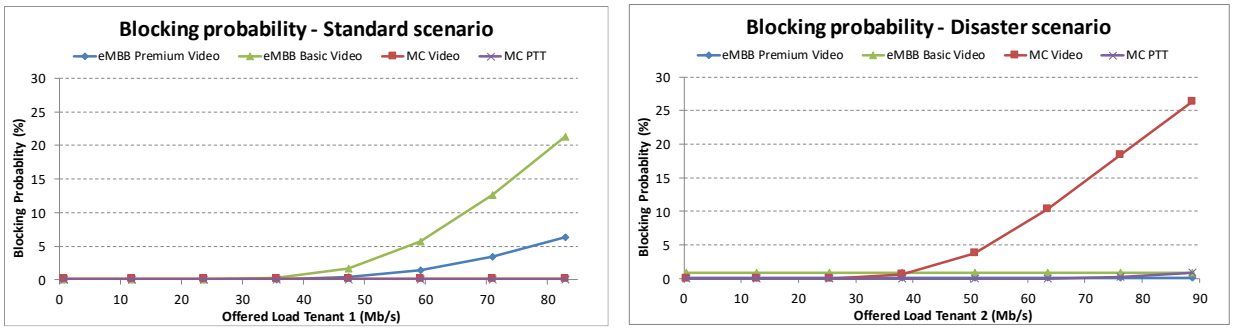
scenario $C_{max,1}$ of the commercial operator is higher whilst in the disaster scenario it is reduced, allowing more resources to be devoted to the public safety operator (i.e., $C_{max,2}$ higher).

The standard and disaster scenarios have been compared based on the analysis of different performance metrics obtained by means of the Markov model. Specifically, the impact of the ARP indicator and the tenant capacity thresholds of the selected AC policy on the system performance under different load conditions is discussed.

Fig. 11 depicts the blocking probabilities for the different services in the standard (Fig. 11a) and the disaster (Fig. 11b) scenarios as a function of Tenant 1 and Tenant 2 offered loads, respectively, understood as the generated traffic load for each of the tenants. For the standard scenario, it can be observed how eMBB services blocking probabilities grow when Tenant 1's offered load is increased. Higher blocking probabilities are observed for the eMBB Basic Video compared to the Premium Video due to the higher ARP value (i.e., lower priority) of the former. In turn, MC services present low blocking probabilities as a result of the low offered load associated with Tenant 2. Specifically, the MC PTT attains almost 0% blocking because the service is granted with the highest priority (i.e., ARP=1). It is also worth noting that, for public safety services, the blocking probabilities remain constant as the eMBB traffic increases, thus reflecting that the AC policy, which establishes capacity thresholds on a per-tenant basis, provides isolation between Tenant 1 and Tenant 2.

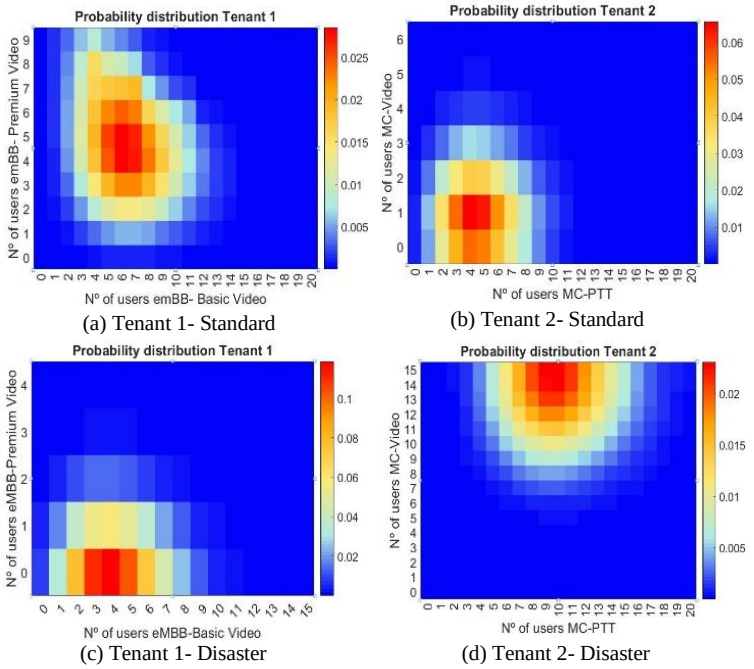
In the disaster scenario, and given that Fig. 11b considers variable load for Tenant 2, the contrary case is found: blocking probability grows for MC services while eMBB services remain invariable. The low blocking probabilities observed for Tenant 1 result from the moderate offered loads, which do not exceed its maximum capacity (i.e., $C_{max,1}$=46.8 Mb/s). Regarding the public safety services, the MC Video can reach substantial blocking in case the traffic grows significantly, whereas the MC PTT service is able to maintain low blocking probabilities.

Blocking probability results can be contrasted by analysing the state probability distribution when setting a fixed load for each Tenant. In particular, Fig. 12 shows the steady-state probability distribution for each of the tenants when Tenant 1's traffic generation rate is set to 0.12 sessions/s (which corresponds to 82.8 Mb/s) in the standard scenario and Tenant 2's traffic generation is set to 0.25 session/s (which corresponds to 88 Mb/s) in the disaster scenario. The axes of the graphs reflect the number of users of each tenant for each represented state probability. For the standard scenario and Tenant 1 (Fig. 12a), the states with highest probability are mostly those with an intermediate number of users for the eMBB Premium video service, while for Tenant 2 (Fig. 12b), the highest probabilities correspond to states with a low number of users in relation to its maximum. This explains that the blocking probability for Tenant 1 is relatively high in Fig. 11a, while for Tenant 2, it is lower.



(a) Standard Scenario (b) Disaster Scenario

Fig. 11. Blocking probability of each service in (a) standard and (b) disaster scenarios.



(a) Tenant 1- Standard (b) Tenant 2- Standard

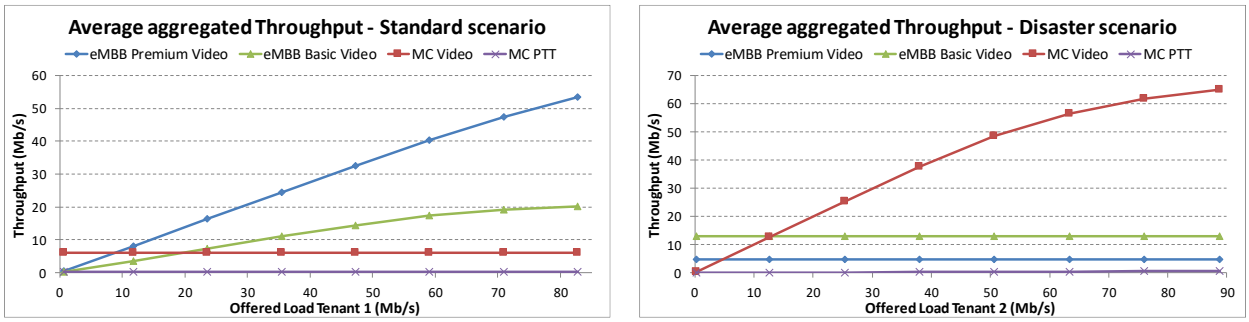
(c) Tenant 1- Disaster (d) Tenant 2- Disaster

Fig. 12. Probability distribution for each of the tenants in the standard scenario (Tenant 1 traffic generation rate set to 0.12 sessions/s) and disaster scenario (Tenant 2 traffic generation rate set to 0.25 sessions/s).

For the disaster scenario, the state space for Tenant 1 (Fig. 12c) is reduced in comparison to the standard scenario case because the $C_{max,1}$ threshold has been set to a lower value (i.e., $C_{max,1}$=46.8 Mb/s compared to $C_{max,1}$=93.6Mb/s in the standard scenario). Therefore, the highest probability is concentrated in states with a low number of users, which is consistent with the low blocking probabilities found in Fig. 11b for Tenant 1. Regarding Tenant 2, the range of possible states is increased as the maximum tenant capacity threshold has been increased to $C_{max,2}$=78 Mb/s. In that case, the highest probability concentrates in states with a large number of users for MC Video service and low number of users for MC PTT service, which results in the large Tenant 2 blocking probability found in Fig. 11b for MC Video and low probabilities for MC PTT.

Although not included graphically, the analysis of the degradation probability reveals that both of the studied scenarios present negligible degradation probabilities (i.e., lower than $3 \cdot 10^{-7}$ for the standard scenario and $2 \cdot 10^{-11}$ for the disaster scenario), implying that the admitted users in the system are provided with its GBR requirements satisfactorily most of the time.

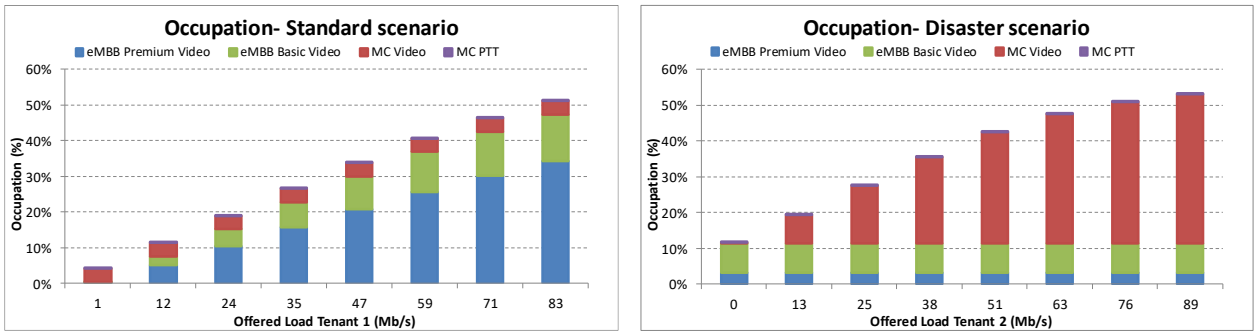
Additionally, the average aggregated throughput per service has been analysed in Fig. 13. For the standard scenario (Fig. 13a), the offered load of eMBB services is increased, so its average throughput for eMBB services grows until achieving high loads, where the system starts overloading. This is directly related to the high blocking probabilities observed for those services in high load conditions in Fig. 11a. In contrast, MC services average throughput remains constant for all loads, as a result of its constant offered load and the low blocking probabilities achieved in the standard scenario. In the case of the disaster scenario (Fig. 13b), eMBB services and MC PTT throughput stay invariable as its corresponding low blocking probabilities (Fig. 11b) whereas MC Video throughput grows until reaching high loads, when the throughput reduces its growing speed as a consequence of the high blocking probabilities obtained for MC-Video in this scenario. By observing the graphs and considering that negligible degradation probabilities are obtained, it can be considered that the increase of offered load in one tenant does not affect the other tenant's throughput.



(a) Standard Scenario (b) Disaster Scenario

Fig. 13 Average aggregated throughput and offered load of each service in (a) standard and (b) disaster scenarios.

Finally, an analysis in terms of the system occupation is conducted for each of the scenarios by considering the average normalised PRB utilisation. For the standard scenario (Fig. 14a), higher PRB utilisation is found for eMBB services than for MC services, which are barely affected when Tenant 1's load is varied. In contrast, higher PRB utilisation is achieved for MC services than for eMBB services in the disaster scenario (Fig. 14b). The reason for this can be found in the variation of the maximum capacity thresholds per tenant for each of the scenarios, as in the standard scenario, higher occupation is allowed to Tenant 1, and in the disaster scenario, the reverse situation is configured. Another relevant effect regarding the results is that the PRB utilisation of MC PTT is truly low, given its low GFBR requirements.



(a) Standard Scenario (b) Disaster Scenario

Fig. 14. Average occupation per tenant and service for the standard and disaster scenarios.

2.7.3. Assessment of the solution with GBR and non-GBR services

While previous sections 2.7.1 and 2.7.2 have analyzed the performance of the Markov model considering only GBR services, this section analyses the solution for scenarios where tenants provide both GBR and non-GBR services. Moreover, the analysis has been conducted under the assumption that the spectral efficiency is modelled as a random variable according to the statistical model proposed for the resource allocation in Section 2.3.4. The scenario under test considers N=2 tenants, referred to as *Tenant* 1 and *Tenant* 2. *Tenant* 1 provides M_1=3 different services while Tenant 2 provides M_2=2 services. For each of the services, the QoS parameters correspond to the used for the model validation in summarised in Table I in terms of the ARP, the PL and the GFBR [5].

The NG-RAN is composed by one gNB with a single cell. Two different radio environments are considered: an UMi and a RMa, which have been configured according to the parameters used for the model validation in Table II. The considered scenarios have been analysed both at state and global system levels. Moreover, a discussion of the impact of introducing a new tenant closes this analysis.

2.7.3.1. State level analysis

In order to analyse the behaviour of the proposed radio resource allocation procedure in L2, two states have been selected and the procedure of Section 2.3.4 has been followed. State #1 comprises $u_{1,1}$=4, $u_{2,1}$=10 and $u_{3,1}$=12 users belonging to Tenant 1 and $u_{1,2}$=6 and $u_{2,2}$=6 belonging to Tenant 2. State #2 is composed of $u_{1,1}$=7, $u_{2,1}$=12 and $u_{3,1}$=12 users belonging Tenant 1 and $u_{1,2}$=9 and $u_{2,2}$=6 users belonging to Tenant 2. Notice that State #2 has higher GBR requirements than State #1. Both states have been evaluated in both UMi and RMa environments, defined in Table II.

Regarding the resource allocation of Tenant 1, Fig. 15, Fig. 16 and Fig. 17 include the pdfs of the assigned resources $f_{a_{x,s,n}}(k)$ in both states and environments for all services of Tenant 1. The impact of the environment can be appreciated: the pdfs for the UMi environment are narrower than the ones for the RMa environment. This reveals that the UMi has better radio propagation conditions than the RMa and less variability in terms of spectral efficiency given the more confined coverage area. In this sense, the peak observed in the obtained pdfs, which is produced by those users having the maximum spectral efficiency available in the system, is much prominent for the pdfs belonging to the UMi scenario.

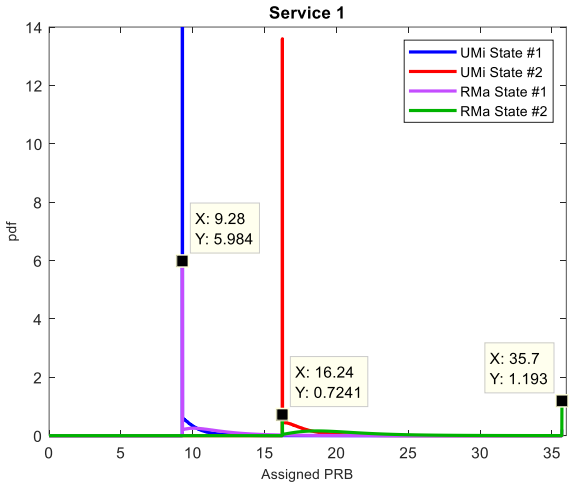


Fig. 15. Pdf of assigned resources to service 1 for State # 1 and State # 2 in UMi and RMa scenarios.

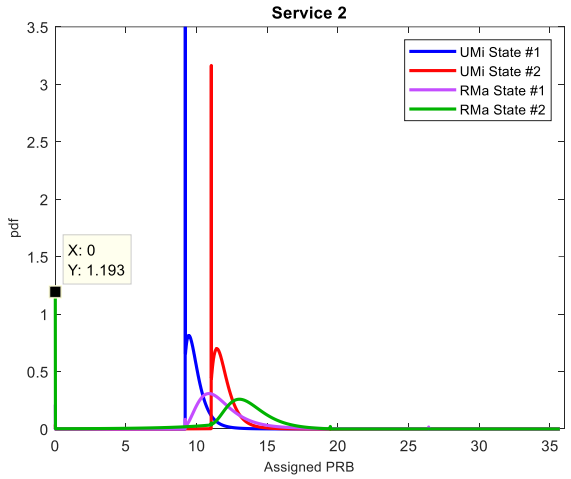


Fig. 16. Pdf of assigned resources to service 2 for State # 1 and State # 2 in UMi and RMa scenarios.

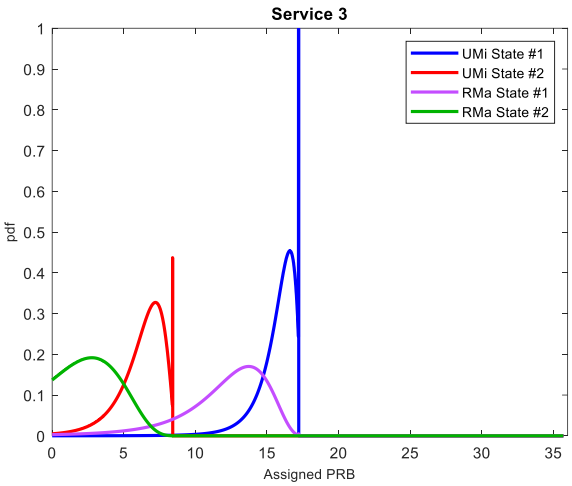


Fig. 17. Pdf of assigned resources to service 3 for State # 1 and State # 2 in UMi and RMa scenarios.

The effect of increasing the number of GBR users can be observed in Fig. 15 and Fig. 16 when comparing the pdfs of States # 1 and # 2 of services 1 and 2, respectively (i.e. $u_{1,1}$=4 and $u_{2,1}$=10 in State # 1 and $u_{1,1}$=7 and $u_{2,1}$=12 in State # 2). As expected, the pdfs in State #2 are shifted to a higher number of assigned PRBs,

as a result of having a greater resource demand. Moreover, a peak at 35.7 PRBs appears for service 1 in State # 2 in the RMa environment, reflecting that there exists a probability that all the resources available for Tenant 1 (i.e., 35.7 PRBs) are assigned to service 1. In this case, given that service 1 has the lowest ARP (i.e. the highest priority), this means that no resources can be assigned to service 2, which is evidenced by the peak at 0 assigned PRBs that appears in Fig. 16.

In the case of service 3, as it is a non-GBR service, it is provided with the remaining resources after the allocation to GBR services (i.e. service 1 and service 2), as shown in Fig. 17. This explains the fact that in State # 1 the pdf is centred in a higher number of resources than in State # 2. Furthermore, the peaks in the pdfs appear at the right hand side instead of the left hand side observed for GBR services, reflecting that service 3 is granted with lower priority and is assigned with the spare radio resources. Table VII presents the state average aggregated PRB utilisation, $\overline{a_{x,s,n}}$, and the state average aggregated throughput, $\overline{Th_{x,s,n}}$, for States # 1 and # 2 and all the services in the system in both UMi and RMa environments. In the case of Tenant 2, given that both services are non-GBR, no requirements in terms of resources are established, so the pdf of the assigned PRBs does not depend on the spectral efficiency or the environment. Consequently, the same values in terms of $\overline{a_{x,s,n}}$ are observed for UMi and RMa environments. From these values, the effect of the proportional sharing constant $\sigma_{s,n}$ is appreciated, which depends on both the PL and the number of users. For State # 1, service 1's average aggregated PRB utilisation is lower than the one obtained for service 2, as the latter has a lower PL (higher priority) and both services have the same number of users. The contrary case is obtained in State # 2, where service 1's average aggregated PRB utilisation is higher than the one obtained for service 2 as a result of the higher number of users of service 1. In terms of the average aggregated throughput of Tenant 2's services, higher throughput is achieved in the UMi scenario thanks to its better propagation conditions.

TABLE VII. AVERAGE AGGREGATED ASSIGNED PRBS AND THROUGHPUT.

Environment		**UMi**		**RMa**	
		$\overline{a_{x,s,n}}$ (PRB)	$\overline{Th_{x,s,n}}$ (Mb/s)	$\overline{a_{x,s,n}}$ (PRB)	$\overline{Th_{x,s,n}}$ (Mb/s)
State #1					
Tenant 1	Service 1	10.05	19.99	11.94	19.99
	Service 2	9.98	19.88	11.85	19.88
	Service 3	15.67	32.02	12.09	22.31
Tenant 2	Service 1	6.55	13.39	6.55	12.09
	Service 2	8.74	17.86	8.74	16.13
State #2					
Tenant 1	Service 1	17.58	35	20.83	34.94
	Service 2	11.94	23.95	12.92	22
	Service 3	6.24	12.74	3	5.53
Tenant 2	Service 1	8.1	16.55	8.1	14.95
	Service 2	7.2	14.7	7.2	13.28

Furthermore, in order to get a deeper insight into the behaviour of the resource allocation procedure from a multi-state perspective, Fig. 18 presents the average aggregated PRBs utilisation $\overline{a_{x,s,n}}$ in the RMa environment for each of the services in the different states, given by the number of users $u_{s,n}$ of each tenant/service. Regarding Tenant 1, Fig. 18a shows that the resource allocation for service 1 is performed

independently of the number of users of service 2. This is because service 1 is the first provided with resources as it has the lowest ARP value (i.e. higher priority). Differently, the resource allocation for service 2 depends on the number of users of service 1 (Fig. 18b), as its PRB allocation is performed after the allocation to service 1. Therefore, the PRB utilisation of service 2 increases when reducing the number of users of service 1. Nevertheless, service 2's PRB utilisation values are lower than in the case of service 1, as the GBR requirement is also lower. Moreover, Fig. 18c proves that service 3 is provided with the remaining resources after the allocation of PRBs of services 1 and 2, so the highest PRB utilisation of service 3 is achieved when the number of users of services 1 and 2 is low. This happens because service 3 is the one with lowest priority (i.e. highest ARP) and is a non-GBR service. Besides, Fig. 18d and Fig. 18e show how the PRBs available to Tenant 2, $N_{th,2}$, are distributed among the non-GBR services 1 and 2 according to the proportional sharing constant $\sigma_{s,n}$. In this sense, both graphs are complementary to each other, i.e. in states with no users of service 1 all the PRBs are allocated to service 2 while in states without users of service 2 all the PRBs are allocated to service 1.

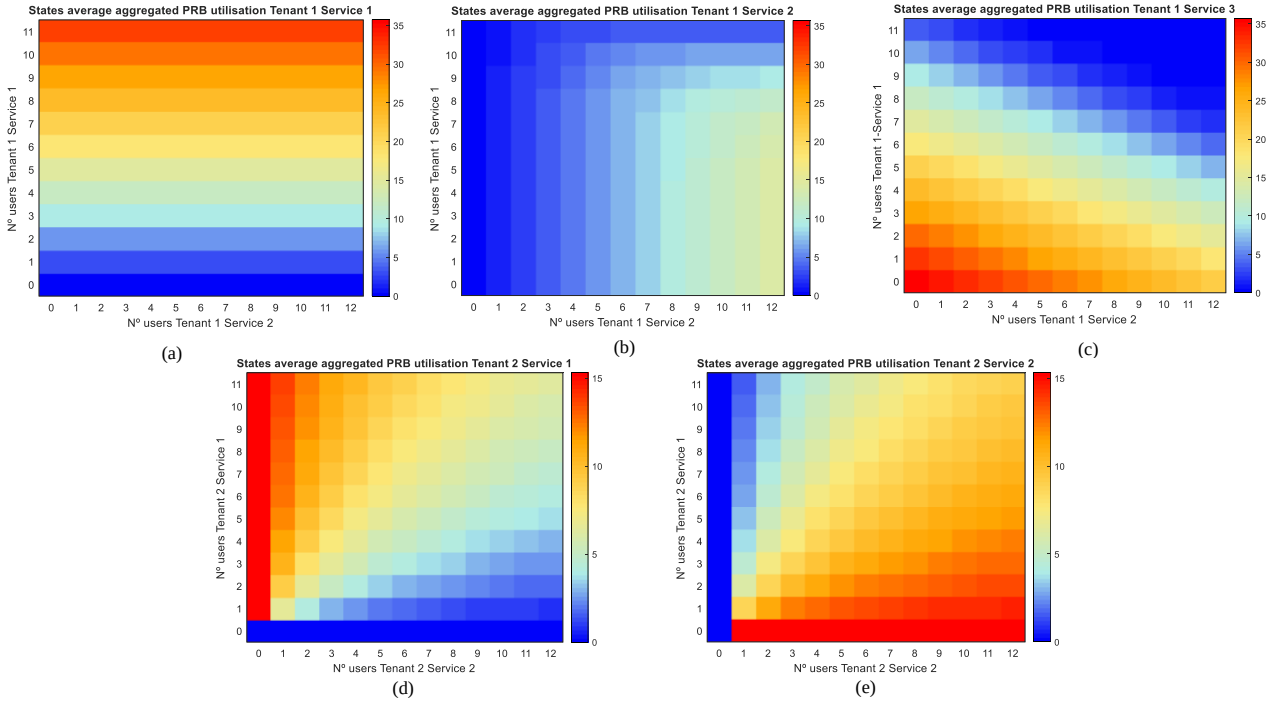


Fig. 18. Average aggregated PRB utilisation in the different states for (a) Service 1 of Tenant 1, (b) Service 2 of Tenant 1, (c) Service 3 of Tenant 1 (d) Service 1 of Tenant 2 and (e) Service 2 of Tenant 2.

2.7.3.2. System level analysis

This section discusses the global system performance in both RMa and UMi scenarios in Table II, focusing on the behaviour of the RRM procedures considered for L3 and L2 and the capability of the overall system to adopt diverse configurations. Results have been generated according to the configuration in Table II, by varying the session generation rate of Tenant 1, which also implies the variation of its GBR services' traffic offered load, and keeping constant the rate of Tenant 2. Traffic offered load by the GBR services of the *n*-th tenant is defined as:

$$\theta_n = \sum_{\substack{s \\ T_{s,n}=0}}^{M_n} GBR_{s,n}\lambda_{s,n}\cdot(1/\mu_{s,n}) \tag{40}$$

To analyse the RRM at L3, Fig. 19 represents the blocking probability of the GBR services in the system (i.e. services 1 and 2 from Tenant 1) in both considered environments as a function of their offered load. For high offered loads, blocking probabilities for service 2 are slightly higher than for service 1 in both environments, exhibiting that it has a higher ARP (i.e. less priority) than service 1. Anyway, the blocking probabilities remain in low values (i.e. less than 1.5%) for the considered offered loads, which are lower than the maximum capacity threshold in both UMi and RMa environments (i.e. $C_{max,1}$=62.6 Mb/s for UMi and $C_{max,1}$=56.2 Mb/s in RMa). In addition, higher blocking probabilities of service 2 are found for the RMa environment. In this case, the admission of a user implies a higher number of required resources, so a lower number of users can be admitted in the system. It is worth pointing out that this effect is not noticed for service 1 due to its higher priority. For non-GBR services, all users are admitted as the maximum number of users $U_{max,s,n}$ is not reached, which is the only constraint considered for them.

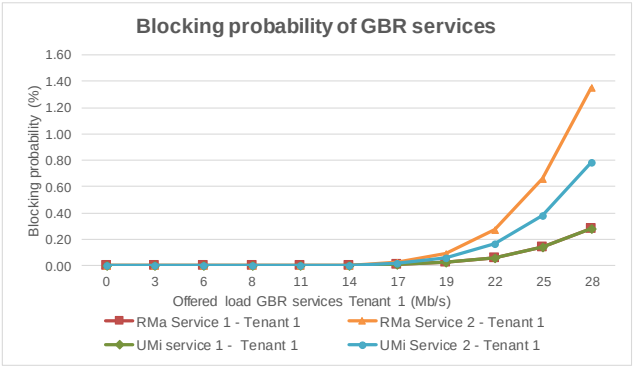


Fig. 19. Blocking probability of GBR services in RMA and UMi environments.

The RRM at L2 is analysed in terms of the system average PRB utilisation, $\overline{a_{s,n}}$, and throughput $\overline{Th_{s,n}}$, both aggregated per service. Fig. 20 represents the system average aggregated PRB utilisation by each of the services for the RMa and UMi environments, as a function of Tenant 1's GBR services offered load. Similar to the previous section, the average aggregated PRB utilisation of GBR services is greater in the RMa scenario as a result of worse propagation conditions, which leads to a higher resource demand. Moreover, the PRB utilisation of GBR services (i.e. service1 and 2 of Tenant 1) is consistent with the configured GBR and ARP values. For instance, the average aggregated PRB utilisation of service 1 of Tenant 1 is greater than the one for service 2 of the same tenant, as service 1 has a larger GBR requirement (i.e. $GBR_{1,1}$=5 Mb/s vs $GBR_{2,1}$=2 Mb/s) and lower ARP (i.e. more priority). Apart from this, when varying the traffic offered load of Tenant 1, the PRB utilisation of services belonging to Tenant 2 remain constant, showing the isolation capability of the resource allocation procedure included in the model.

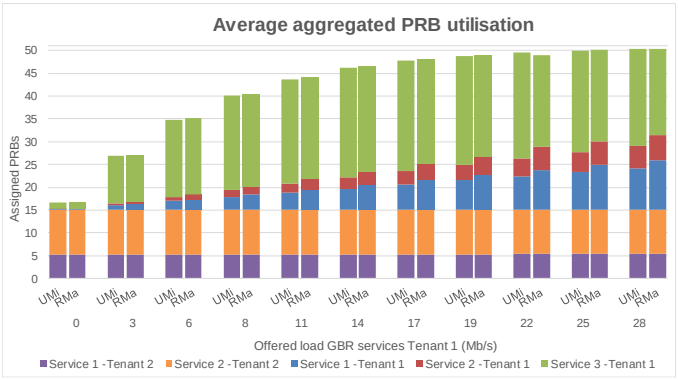


Fig. 20. Average assigned PRBs aggregated per service in the RMa and UMi environments.

In terms of the system average aggregated throughput, $\overline{Th_{s,n}}$, GBR services present equivalent results to the ones obtained for the PRB utilisation. For GBR services, the obtained average aggregated throughput is proportional to the GBR value and the average number of users in the system. This proportionality is achieved thanks to the low degradation probabilities that are obtained in the considered scenarios (maximums of 0.001% for UMi and of 2% in the RMa).

However, the mentioned proportionality is not found for non-GBR services, so its throughput dynamics deserve a deeper analysis. Fig. 21 shows the behaviour of the average aggregated throughput in both RMa and UMi scenarios for non-GBR services, this time as a function of Tenant 1's session generation rate, which embraces all Tenant 1's services. For all non-GBR services, higher average aggregated throughput is given in the UMi scenario provided its better propagation conditions. This occurs in spite that the average PRB utilisation of all the services of Tenant 2 in Fig. 20 is the same for both environments and, in the case of service 3 of Tenant 1, it is larger for the RMa environment. This reflects that the throughput metric for non-GBR does not only depend on the number of assigned resources but also on the propagation conditions in each scenario.

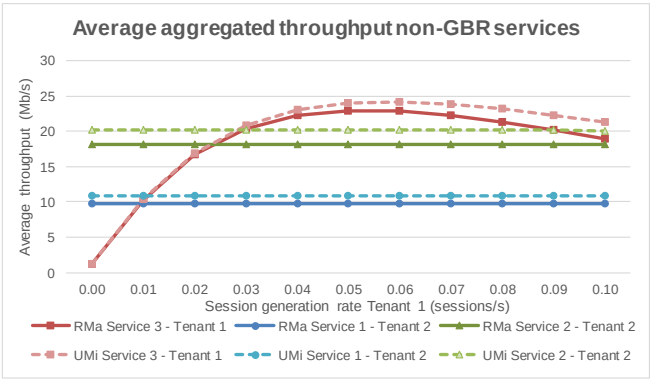


Fig. 21. Average aggregated throughput to non-GBR services in the RMa and UMi environments.

Besides, as a result of the isolation capability achieved at layer 2, Fig. 21 also shows that the aggregated average throughput of Tenant 2 services remains constant when increasing the session generation rate of Tenant 1. The effect of the PL is also noticeable as higher throughput is provided to service 2, which has the lowest PL (i.e. higher priority). Instead, the throughput of service 3 of Tenant 1 experiences a different trend depending on the session generation rate of this tenant. Specifically, the average throughput of this

service increases with the session generation rate up to approximately 0.05 sessions/s. The reason for this is that increasing the session rate raises the probability of having at least one user of this service that can exploit all the PRBs left by GBR users. However, for traffic generation rates higher than 0.05 sessions/s, the average throughput starts to decrease as there are less available resources to service 3, which is caused by a higher resource requirement of GBR services.

2.7.3.3. Analysis of the introduction of a new tenant

One key feature of network slicing is the flexibility to create, re-configure and release network slices. The creation of a network slice implies the allocation of a suitable capacity to support the traffic of a new tenant. In case that the existing tenants do not consume all the capacity in the system, the L3 maximum capacity $C_{max,n}$ and L2 maximum number of PRBs $N_{th,n}$ of the new slice can be configured to provide the required capacity from the remaining part. If no capacity is available, as it is the case in the example of previous section, where Tenant 1 and Tenant 2 already consume 100% of the PRBs of the gNB, the simplest approach would be to extend the system capacity e.g., through extending the assigned bandwidth. In case that this is not feasible, due to e.g. hardware constraints or spectrum limitation, the re-configuration of the existing slices so that the actual capacity is redistributed among all the tenants could be explored. In order to illustrate these approaches, this section discusses the addition of a new tenant in the considered scenario.

Let us assume that the new tenant, denoted as *Tenant 3*, provides two GBR services with $GBR_{1,3}$=1 Mb/s and $GBR_{2,3}$=0.5 Mb/s, both of them with the same priority $ARP_{1,3}$=$ARP_{2,3}$=2. The average session generation rate of this tenant is 0.07 sessions/s, the average session duration is 120s, and 70% of sessions belong to service 1 and 30% to service 2.

The first solution to create the new slice, denoted as *Solution A*, increases the PRB availability in the gNB, by increasing the cell bandwidth from 20 to 25 MHz, which results in an increase of available PRBs from N_{ava}=51 to N_{ava}=65 PRBs [46]. Then, Tenant 3 is configured to use part of this additional capacity by configuring the values of the thresholds L3 maximum capacity $C_{max,3}$ and L2 maximum number of PRBs $N_{th,3}$ as in Table V. The corresponding thresholds of Tenants 1 and 2 remain unchanged with respect to Table II. In contrast, the second solution, denoted as *Solution B*, re-configures the values of $C_{max,n}$ and $N_{th,n}$ for the existing tenants. Specifically, given that Tenant 2 only carries non-GBR services, the selected re-configuration consists in reducing the value of $N_{th,2}$ without modifying the slicing thresholds of Tenant 1, as seen in Table VIII. For the new Tenant 3 the same thresholds as in *Solution A* are used.

TABLE VIII. SLICING THRESHOLDS FOR THE PROPOSED SOLUTIONS TO INCLUDE TENANT 3.

Tenant id (*n*)	N_{ava}	**RMa**		**UMi**	
		$C_{max,n}$	$N_{th,n}$	$C_{max,n}$	$N_{th,n}$
Solution A: Increase PRB availability					
1	65 PRBs	62.4 Mb/s	35.7 PRBs	56.4 Mb/s	35.7 PRBs
2		N/A (only non-GBR services)	15.3 PRBs	N/A (only non-GBR services)	15.3 PRBs
3		20.8 Mb/s	10 PRBs	18.8 Mb/s	10 PRBs
Solution B: Re-configure slicing thresholds					
1	51 PRBs	62.4 Mb/s	35.7 PRBs	56.4 Mb/s	35.7 PRBs
2		N/A (only non-GBR services)	5.1 PRBs	N/A (only non-GBR services)	5.1 PRBs
3		20.8 Mb/s	10 PRBs	18.8 Mb/s	10 PRBs

Regarding RRM L3, the blocking probability $P_{s,n}^{b}$ of all the GBR services in the system has been obtained for *Solution A* and *Solution B* when the session generation rate of Tenant 1 is 0.1 sessions/s. Results are presented in Table IX. Since the value of $C_{max,1}$ and $C_{max,3}$ is the same in both solutions, the obtained blocking probabilities for all GBR services of Tenants 1 and 3 are also the same. It is observed that the blocking probabilities of Tenant 1 do not change with the introduction of Tenant 3, i.e. they are the same as in Fig. 19. This shows that the AC algorithm provides isolation in the admission of GBR users. Regarding Tenant 3, although blocking probabilities are low for both UMi and RMa, the lowest blocking probabilities are achieved once again in the UMi environment. Moreover, blocking probabilities of service 1 are higher than those of service 2 although both services have the same ARP value. The reason is that the traffic offered for service 1 of Tenant 3 is higher than that of service 2.

TABLE IX BLOCKING PROBABILITY OF GBR SERVICES FOR SOLUTION A AND B.

Tenant (*n*)	**Service (*s*)**	**Blocking probability $P_{s,n}^{b}$ (%)**			
		Solution A		***Solution B***	
		RMa	**UMi**	**RMa**	**UMi**
1	1	0.282	0.282	0.282	0.282
	2	1.351	0.788	1.351	0.788
3	1	0.014	0.002	0.014	0.002
	2	0.011	0.001	0.011	0.001

The analysis of the proposed solutions for including Tenant 3 has also been analysed in relation to RRM at L2 by means of the PRB utilisation $\overline{a_{s,n}}$ and the average aggregated throughput $\overline{Th_{s,n}}$, also for a session generation rate of Tenant 1 equal to 0.1 sessions/s. Fig. 22 contains the comparison of the PRB utilisation values obtained for each of the services for *Solutions A* and *B* in the RMa and UMi environments. It is observed that the PRB utilisation of Tenant 1 and Tenant 3 is the same for both solutions, because the thresholds $N_{th,1}$ and $N_{th,3}$ of these tenants are the same. Instead, when comparing the PRB utilisation of services 1 and 2 of Tenant 2 in *Solution A* and *B*, significant differences are obtained. In fact, the PRB utilisation for *Solution B* is 66.6% lower than the one obtained for *Solution A* in both RMa and UMi, which matches the reduction of Tenant 2 $N_{th,2}$ threshold (i.e., from $N_{th,2}$=15.3 PRBs in *Solution A* to $N_{th,2}$=5.1 PRBs in *Solution B*). In terms of the PRB utilisation of the Tenant 3 services, higher utilisation is obtained for

service 1 than for service 2, due to the fact that the GBR requirement of the former is twice the GBR requirement of the latter and the session generation of service 1 is also higher.

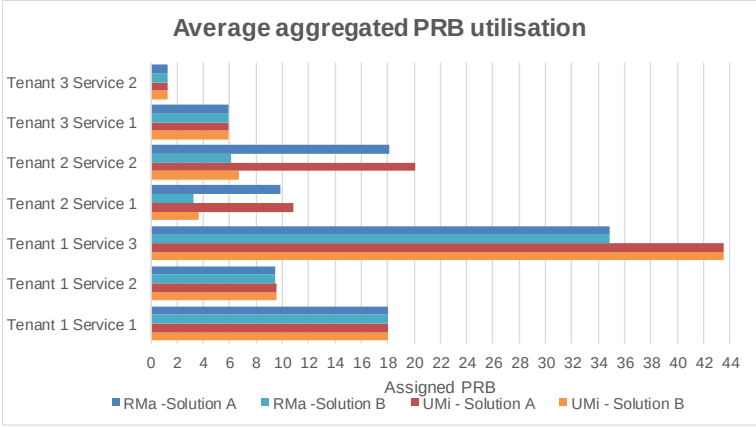


Fig. 22. Average assigned PRBs aggregated per service in the RMa and UMi environments for Solution A and B.

The decrease of Tenant 2 PRB utilisation has as a consequence a reduction in its aggregated throughput $\overline{Th_{s,n}}$, as shown in Fig. 23, which shows the average aggregated throughput for non-GBR services with *Solutions A* and *B*. The performance of service 3 of Tenant 1 is the same for both solutions given that Tenant 1 thresholds remain unchanged, while Tenant 2 services present a 66.6% reduction with *Solution B* in comparison to *Solution A*, which is the same decrement observed in the PRB utilisation. Therefore, the realization of *Solution B* may be subject to a renegotiation of the Service Level Agreement terms with Tenant 2.

Regarding GBR services, the aggregated average throughput is shown in Table VII. In this case, as GBR values are provided almost always and low degradation probabilities are achieved for all services, the resulting throughput $\overline{Th_{s,n}}$ is approximately the product of the average number of users $\overline{U_{s,n}}$ of each GBR service and the GBR value. Since the $C_{max,n}$ of Tenant 1 and Tenant 3 have not been changed for *Solutions A* and *B*, the obtained $\overline{U_{s,n}}$ and $\overline{Th_{s,n}}$ of each service is the same, so Table VII does not present separate results for each solution. Besides, it is also observed that very small differences are obtained in the results of the UMi and RMa environments.

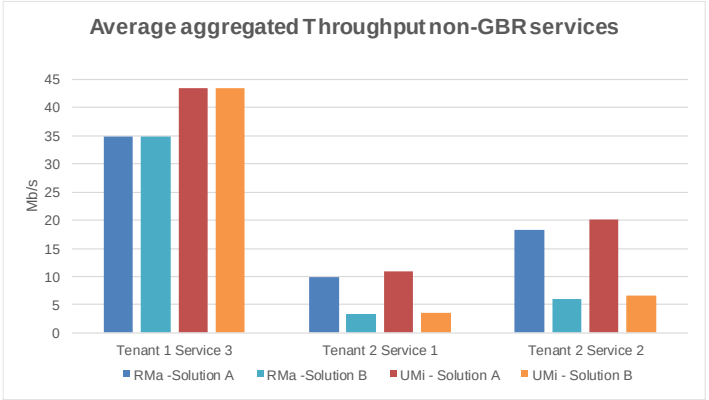


Fig. 23. Average aggregated throughput to non-GBR services in the RMa and UMi environments for Solution A and B.

TABLE X. AVERAGE NUMBER OF USERS AND AVERAGE AGGREGATED THROUGHPUT FOR GBR SERVICES IN THE SYSTEM.

Tenant id (n)	Service id (s)	RMa		UMi	
		$\overline{U_{s,n}}$ (users)	$\overline{Th_{s,n}}$ (Mb/s)	$\overline{U_{s,n}}$ (users)	$\overline{Th_{s,n}}$ (Mb/s)
1	1	3.60	18.01	3.60	18.02
	2	4.74	9.41	4.74	9.53
3	1	5.91	5.89	5.92	5.91
	2	2.52	1.26	2.53	1.27

2.8. Summary

This chapter has presented a Markov model that characterises the resource sharing in RAN slicing scenarios, where multiple tenants can provide GBR and non-GBR services. The model includes diverse RRM functions in terms of admission control at layer 3 and radio resource allocation at layer 2. In terms of admission control, which determines the transition probabilities between the different states in the model, a slice-aware admission control policy has been selected. The model has also been provided with a slice-aware resource allocation procedure. This procedure has been initially defined in average terms but has been upgraded to variable propagation conditions by deriving the pdfs of both the required and assigned resources according to the service's QoS parameters.

The Markov model has been validated by contrasting the its performance with that achieved through system level simulations under different environments and load levels. The results have shown that the model is suitable to obtain reliable results to study RAN slicing when considering diverse environments in terms of cell deployment (cell radius, transmitted power, etc.) and different traffic loads, given the low percentage errors obtained (i.e. maximum relative errors of 3%) when comparing the model's results with the ones obtained with the system level simulator.

Moreover, the effect of the considered RRM functions has been studied by evaluating different performance metrics (blocking probability, degradation probability, throughput, occupation, etc.) in a wide range of scenarios. The scenarios under evaluation have ranged in the considered services types (i.e., GBR and non-GBR services types), the QoS requirements, cell deployment and traffic load conditions and have also covered relevant use cases envisaged for 5G that embrace eMBB and MC services. The obtained results have revealed that:

(i) The considered admission control policy and resource allocation procedure are able to achieve isolation between the different slices, so that overload situations in one slice do not affect the performance of the other slices while preserving the maximum capacity allowed to each of the slices.

(ii) ARP priorities are respected by providing better performance to those GBR services with lower ARP (higher priority).

(iii) The proposed admission control policy provides negligible degradation rates to GBR services, which implies that the requested GBR values are provided to the admitted users in the system.

(iv) Non-GBR services are provided with the remaining resources after the allocation to GBR services according to its priority level, so that better performance is given to those non-GBR services with higher priority (lower priority level);

(v) The introduction of new tenants into the system can be performed by re-configuring the maximum capacity and maximum PRBs to be provided to each of the tenants, although the re-configuration of these values may impact on the performance of already operating tenants if the total amount of PRBs is not modified.

(vi) The model is able to satisfactorily capture the radio propagation effects, enabling the analysis of different performance metrics in 5G environments of interest.

Overall, the proposed framework has proven to be an appropriate platform for the evaluation and the analytical characterisation of RAN slicing aspects in wide range of use case scenarios, allowing the establishment of relationships between the different dimensions of the RAN slicing problem.

Chapter 3. A data analytics-based Slice Admission Control methodology

3.1. Introduction

This chapter addresses the problem of Slice Admission Control (SAC) by proposing a novel methodology for estimating the required radio resources by a RAN slice based on data analytics. The estimated required radio resources are used by the SAC to determine the acceptance or rejection of a new RAN slice creation request, so that the Service Level Agreement (SLA) established between the service provider (e.g. the Infrastructure Provider (InP) and the customer (e.g. a Mobile Virtual Network Operator (MVNO)) is ensured. The proposed approach is based on the extraction of data analytics information from the cell performance measurements collected at the management plane and is fully aligned with the 3GPP vision for Next Generation RAN (NG-RAN) management exploiting Management Data Analytics Function (MDAF) and RAN Slicing Management Function (RSMF). The methodology relies on an extension of the statistical model developed in Section 2.3.4 that allows deriving the pdf of the required and assigned resources by a RAN slice, considering both Guaranteed Bit Rate (GBR) and non-GBR services.

The methodology has been evaluated under diverse cell environments, including simulation environments based on 3GPP reference models as well as some representative cells and performance measurements extracted from a real LTE commercial network. Moreover, the methodology has been applied to a Mission Critical (MC) scenario, considered by the 5G ESSENCE project.

The rest of the chapter is organized as follows. Section 3.2 provides the literature review of the SAC problem. Section 3.3 identifies the problem of SAC and Section 3.4 describes the considered functional framework for analytics-based RAN slicing management encompassing SAC. Then, Section 3.5 presents the proposed methodology to estimate the amount of required resources by a RAN slice whereas Section 3.6 includes the results obtained for different cell deployment and load scenarios. Finally, Section 3.7 summarizes the conclusions of this chapter.

3.2. Literature review

The slice admission control (SAC) problem in 5G has been addressed at different levels in the literature such as in [49], which identifies SAC as one of the stages of slice creation and establishes the information required for this function or in [35], which proposes a Markov model for asynchronous SAC considering slice required resources as general units with an assigned cost. Similarly, [50] presents an economic model

that evaluates the profit of the MNO for accepting a certain slice and [51] maximises the time-average revenue by a SAC for delay-tolerant slices. In relation to RAN slicing, RAN SAC for slice requests that need to support a given number of users for a certain time is proposed in [24] and [25], aiming at optimising the infrastructure providers' revenue, and in [26], which optimises the network utilisation by incorporating traffic forecasting capabilities. In the above-mentioned works, the amount of required radio resources for a given slice is either taken as an input parameter or computed assuming a fixed capacity per cell. However, the variability in the radio propagation and interference conditions experienced by the users when operating in different cells, the cell location (e.g., indoor, urban, rural environments), the cell deployment (e.g., cell radius, transmitted power, frequency of operation, modulation, antennas, etc.) together with the diverse users' mobility patterns is considered here, given its high impact on the cell's capacity and, consequently, the amount of required radio resources to support a given RAN slice.

3.3. Problem identification

A multi-tenant scenario with a common NG-RAN infrastructure operated by a InP is considered [4]. The NG-RAN infrastructure is composed of multiple cell sites with different deployment characteristics. The InP provides each tenant with a RAN Slice Instance (RSI) encompassing a set of cell sites during a certain time and is responsible of the Lifecycle Management (LCM) of the different RSIs deployed in the NG-RAN infrastructure. Each tenant *n* can provide Guaranteed Bit Rate (GBR) services, which ensure its users with a bit rate value GBR_n (also referred to as Guaranteed Flow Bit Rate (GFBR) in 5G New Radio's (NR) terminology), and non-GBR services. The SLA established between the InP and the *n*-th tenant comprises the following system level indicators:

- **Maximum number of GBR and non-GBR services per cell:** each cell has to support $u_{GBR,n}$ GBR flows and $u_{nGBR,n}$ non-GBR flows.
- **Degradation probability of GBR services:** The percentage of session time that GBR services may experience transfer bit rates lower than its requirement GBR_n must not exceed a maximum value, defined as a degradation probability $P_{d,GBR,n}^{max}$.
- **Average throughput of non-GBR services:** non-GBR flows with buffered traffic need to be provided with a minimum average throughput $\overline{Th_{nGBR,n}^{min}}$ measured over a certain averaging window time.

When a tenant requires to be allocated with a new RSI, the SAC function determines its acceptance or rejection based on the estimation of the required radio resources by the RSI and on the resource occupancy in the NG-RAN infrastructure. Assuming 5G NR technology, radio resource requirements are expressed in terms of the number of Physical Resource Blocks (PRB). If the required PRBs can be utilised in the NG-RAN infrastructure without impacting on the SLAs of already established RSIs, the SAC function admits the new RSI. Otherwise, the RSI is rejected.

3.4. Functional framework for Analytics-based RAN slicing management

Fig. 24 depicts the functional framework for the management of network slicing in a NG-RAN infrastructure leveraging data analytics functions for supporting RAN SAC.

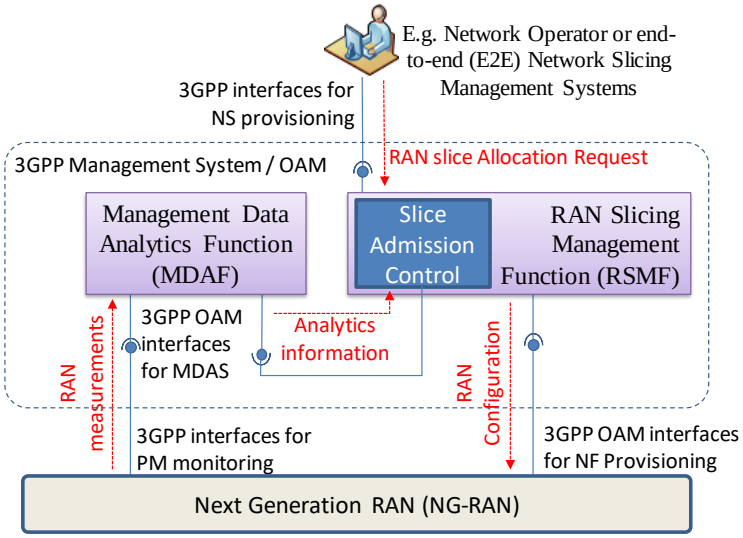


Fig. 24.Proposed functional framework for analytics-based RAN SAC.

The core functionality consists of a set of management functions, collectively referred to as *RAN Slicing Management Function (RSMF).* The *RSMF* is in charge of the LCM of the RSIs, including the admission control of new RSIs to be allocated in the NG-RAN infrastructure and its further creation, modification and termination. In this regard, the *RSMF* is expected to expose a set of management services for provisioning, performance monitoring and fault management of the RSIs. Accordingly, and in line with the terminology and service-based management concepts adopted by 3GPP, the RSMF plays the role of a producer of management services that can be accessed by one or multiple management service consumers. On the other hand, in order to interact with the underlying infrastructure components and carry out the SAC and LCM of RSIs, the RSMF has to be able to consume the services provided by the diverse management systems specific to each of the function types composing the NG-RAN infrastructure (i.e. gNB functions, Network Function Virtualisation Infrastructure (NFVI), Radio Frequency (RF) systems and Transport Network (TN) nodes). In this respect, for the management of the gNB Network Functions (NFs), the 3GPP specifications include the Network Resource Model (NRM) definitions that, apart from attributes used for configuring the operation of the NR cells (e.g. cell identifiers, channel frequencies), contains also attributes for the configuration of the network slicing behaviour of gNB NFs [52].

As part of the 3GPP management system, the Management Data Analytics Function (MDAF) is the component in charge of providing Management Data Analytics Services (MDAS). For example, the MDAF can retrieve Operations, Administration and Management (OAM) data from the network functions and produce management analytics information that can be used to recommend appropriate management actions to the network operator. In turn, with the advances in Machine Learning (ML) and big data technologies for the implementation of MDAS, the 5G management functions might have access to a richer set of MDAS

information extracted from raw data collected over days, weeks, months and even beyond. In this sense, the data analytics services delivered by the MDAS are proposed to be used here for supporting the SAC function at the RSMF. Specifically, the MDAF will provide the statistical characterisation of the spectral efficiency in the different cells of the NG-RAN, which can be derived from OAM performance measurements collected from gNB NFs [41]. This information will be consumed by the RSMF to estimate the PRB requirements of the new RSIs.

3.5. Estimation methodology

This section describes the proposed methodology for the estimation of the required PRBs to fulfil the SLA of a new RSI admission request associated to tenant n under the described management framework. This number of required PRBs is denoted as $N_{th,n}$. Given that the new RSI may operate in diverse cell sites with variety of deployment characteristics and, consequently, users experience different propagation conditions, the proposed methodology is performed on a per cell-basis using the cell characterisation provided by the MDAF. Given that requirements of GBR and non-GBR services are of different nature, the methodology assumes that $N_{th,n}$ is decoupled into $N_{GBR,n}$ and $N_{nGBR,n}$, which are the required PRBs to fulfil GBR and non-GBR requirements, respectively, so that $N_{th,n} = N_{GBR,n} + N_{nGBR,n}$.

Considering the above, and taking into account the SLA terms specified in Section 3.3, the proposed methodology is devised as a three-step procedure:

- **Step 1 – Data analytics:** The MDAF gathers information from the cell environment, processes it and provides it to the RSMF.
- **Step 2 – Resource estimation of GBR services:** $N_{GBR,n}$ is determined by considering that the $u_{GBR,n}$ GBR flows need to be always provided with GBR_n while respecting a maximum degradation probability $P^{max}_{d,GBR,n}$.
- **Step 3 – Resource estimation of non-GBR services:** Additionally to the obtained $N_{GBR,n}$, some extra PRBs $N_{nGBR,n}$ may be needed to fulfil the average $\overline{Th^{min}_{nGBR,n}}$ requirement of the $u_{nGBR,n}$ non-GBR flows.

In the following, more insights into the three steps of the proposed methodology are provided, which rely on an extension of the statistical model developed for resource allocation in Section 2.3.4.

3.5.1. Step 1: Data analytics

The MDAF is in charge of providing the cell characterisation to the RSMF for the estimation of the required PRBs. The deployment and configuration of a certain cell within the NG-RAN infrastructure (radius, transmitted power, location, frequency operation, etc.) and the propagation conditions that users experience in different cell locations result in fluctuations in the spectral efficiency S_{eff} (bits/s/Hz). Therefore, S_{eff} can be treated as a random variable, which is used for the characterisation of each cell environment.

The probability density function (pdf) of the S_{eff}, $f_{Seff}(y)$, is obtained by the MDAF based on the gathering and averaging of samples of the wideband CQI distribution computed in each cell, according to the procedure described in Section 2.3.4.1. Based on $f_{Seff}(s)$, the MDAF computes the following metrics to be provided to the RSMF carrying out the subsequent steps 2 and 3 of the PRB estimation process: (a) average spectral efficiency $\overline{S_{eff}}$ of the cell, (b) the pdf of the random variable $Y=1/(S_{eff} \cdot B)$, denoted as $f_Y(y)$, and computed according to equation (8), where B is the PRB bandwidth.

3.5.2. Step 2: Resource estimation of GBR services

The required PRB estimation performed in the RSMF firstly computes the required $N_{GBR,n}$ to fulfil the GBR services requirements. To achieve GBR_n, each GBR flow requires a number of PRB $N_{req,GBR,n}$, which is obtained by equation (6) and it is a random variable since it depends on Y. The pdf of $N_{req,GBR,n}$ is derived by equation (9) and, based on this pdf and assuming that each user experiences independent propagation conditions, the pdf of the aggregate number of required PRBs $r_{GBR,n}$ by the $u_{GBR,n}$ GBR flows of the n-th tenant, $f_{r_{GBR,n}}(r)$ can be computed by equation (10).

Considering that the degradation probability is given by the probability of requesting more PRBs than the available ones (i.e. $r_{GBR,n} > N_{GBR,n}$), the $N_{GBR,n}$ value that fulfils the $P^{max}_{d,GBR,n}$ requirement is taken as the minimum value that satisfies the following condition:

$$P^{\max}_{d,GBR,n} = \int_{N_{GBR,n}}^{\infty} f_{r_{GBR,n}}(k)dk \tag{41}$$

3.5.3. Step 3: Resource estimation of non-GBR services

Once $N_{GBR,n}$ has been obtained, the following step is to determine if the tenant n is required to be provided with some additional PRBs $N_{nGBR,n}$ to satisfy the requirements of non-GBR services, which need to be provided with $\overline{Th^{min}_{nGBR,n}}$. In this sense, the average throughput of a non-GBR flow is defined as follows:

$$\overline{Th_{nGBR,n}} = \frac{\overline{a_{nGBR,n}} \cdot \overline{S_{eff}} \cdot B}{u_{nGBR,n}} \tag{42}$$

where $\overline{a_{nGBR,n}}$ is the average aggregated assigned PRBs to all the $u_{nGBR,n}$ non-GBR flows of tenant n and $\overline{S_{eff}}$ has been provided by the MDAF. Then, in order to fulfil the minimum throughput requirement $\overline{Th_{nGBR,n}} \geq \overline{Th^{min}_{nGBR,n}}$, the minimum average aggregated assigned PRBs that should be available to non-GBR flows of tenant n, $\overline{a^{min}_{nGBR,n}}$,is given by:

$$\overline{a^{\min}_{nGBR,n}} = \frac{\overline{Th^{\min}_{nGBR,n}} \cdot u_{nGBR,n}}{\overline{S_{eff}} \cdot B} \tag{43}$$

To make this computation, the average aggregated assigned PRBs $\overline{a_{nGBR,n}}$ to non-GBR flows is derived by equation (25), which relies on the computation of the pdf of the aggregated assigned PRBs to all non-GBR

flows of tenant n, $f_{a_{nGBR,n}}(k)$. The computation of $f_{a_{nGBR,n}}(k)$ assumes that the allocation criteria allocates the remaining PRBs after the allocation of GBR flows to the non-GBR flows and depends on the pdf of the aggregated assigned PRBs to all GBR flows, $f_{a_{GBR,n}}(k)$, given by:

$$f_{a_{GBR,n}}(k) = f_{r_{GBR,n}}(k) \cdot H(k, N_{GBR,n}) + \delta(k - N_{GBR,n}) \cdot \int_{N_{GBR,n}}^{\infty} f_{r_{GBR,n}}(t)dt \quad (44)$$

where $\delta(\cdot)$ is the Dirac delta function and $H(x,y)$ is defined according to equation (17). Note that equation (44) corresponds to a simplified version of equations for the case of not considering ARP priorities among GBR services. In this equation, $f_{a_{GBR,n}}(k)$ equals $f_{r_{GBR,n}}(k)$ for all those values lower or equal than $N_{GBR,n}$, as each flow gets the required resources (i.e. $a_{GBR,n} = r_{GBR,n}$). In turn, when the aggregated required PRBs exceed $N_{GBR,n}$ (i.e. $r_{GBR,n} > a_{GBR,n}$) the aggregated assigned PRBs are $a_{GBR,n} = N_{GBR,n}$, as no more PRBs are available. Consequently, all $f_{r_{GBR,n}}(k)$ values where $r_{GBR,n} > N_{GBR,n}$ are cumulated in $a_{GBR,n} = N_{GBR,n}$. Once $f_{a_{GBR,n}}(k)$ has been obtained, $f_{a_{nGBR,n}}(k)$ can be derived by considering the remaining PRBs after the allocation of GBR flow (i.e., $N_{GBR,n} - a_{GBR,n}$). Accordingly, $f_{a_{nGBR,n}}(k)$ is defined as:

$$f_{a_{nGBR,n}}(k) = f_{a_{GBR,n}}(N_{GBR,n} + N_{nGBR,n} - k) \quad (45)$$

where $N_{nGBR,n}$ is added to $N_{GBR,n}$ in order that $f_{a_{nGBR,n}}(k)$ satisfies the $\overline{a^{min}_{nGBR,n}}$ requirement, so that $\overline{a_{nGBR,n}} \geq \overline{a^{min}_{nGBR,n}}$. Then, the additional PRBs, $N_{nGBR,n}$, can be obtained by performing the iterative procedure in Algorithm 1, where $N_{nGBR,n}$ is progressively increased in steps of Δ until fulfilling the $\overline{a^{min}_{nGBR,n}}$ requirement.

Algorithm 1 Iterative procedure for $N_{nGBR,n}$ computation

```
Initialisation
Set N_nGBR,n=0 and the PRB iteration step Δ
Iteration
  Obtain the pdf f_anGBR,n(k) by (45)
  Calculate a_nGBR,n by (25)
  If (a_nGBR,n < a^min_nGBR,n)
    Upgrade N_nGBR,n= N_nGBR,n+Δ
  Else
    Exit iteration loop
  End if
End
```

3.6. Performance evaluation

In this section, the performance of the proposed estimation methodology of the PRB requirements for new RSI admission requests has been evaluated. First, the methodology has been evaluated in diverse cell environments, including simulated-based cell environments and cells selected from a real LTE deployment dataset. Secondly, the methodology has been applied to a relevant 5G use case scenario to estimate the PRB requirement of a RAN slice supporting a Mission Critical Push To Talk (MC PTT) service during an emergence.

3.6.1. Evaluation of the solution in diverse cell environments

3.6.1.1. Considered scenario

The scenario under test considers that two new tenants, referred to as *Tenant 1* and *Tenant 2*, request the allocation of a RSI in the NG-RAN infrastructure of an InP. Both tenants intend to use the RSI to provide GBR and non-GBR services with the SLA parameters summarised in Table XI. Note that different values of the maximum number of flows per cell are considered in the assessment.

The analysis has been performed in the following selected types of cells: an Indoor Hotspot (InH), a Urban Micro-cell (UMi), a Urban Macro-cell (UMa) and a Rural Macro-cell (RMa) [45]. The cell configuration parameters are summarised in Table XII. The S_{eff} distribution of these cells configurations have been obtained by simulating users at different random positions uniformly distributed within the cells and extracting their spectral efficiency.

Additionally, a LTE network deployment in a large European city has been considered in the study. From the available dataset, which includes the CQI distribution of about 60 cells, three representative cells, referred to as cell#1, cell#2 and cell#3, have been selected. These cells correspond to a UMa-like environment.

TABLE XI. SERVICES PER TENANT.

Service Type	SLA/QoS	Tenant 1	Tenant 2
GBR	GBR_n	10 Mb/s	5 Mb/s
	$P^{max}_{d,GBR,n}$	1%	5%
	$u_{GBR,n}$	From 20 flows to 50 flows.	
Non-GBR	$\overline{Th^{min}_{nGBR,n}}$	1 Mb/s	0.5 Mb/s
	$u_{nGBR,n}$	From 20 flows to 50 flows.	

TABLE XII. CELL ENVIRONMENTS CONFIGURATION.

Environment	InH	UMi	UMa	RMa
ISD (Inter-Site distance)	20 m	200 m	500 m	1732 m
gNB height	3 m	10 m	25 m	35 m
UE height	1 m	1.5 m	1.5m	1.5 m
Minimum gNB-UE distance	0 m	10 m	35m	35 m
Average building height (*h*)	-	-	-	5 m
Average Street width (*W*)	-	-	-	20 m
Path Loss and Shadowing model	Model of Sec. 7.4 of [45]			
Shadowing standard deviation in LOS	3	4	4	4
Shadowing standard deviation in NLOS	8.03	7.82	6	8
Frequency	24.9 GHz	3.6 GHz	3.6 GHz	704 MHz
gNB transmitted power (dBm/PRB)	-3 dBm	11 dBm	22 dBm	20 dBm
gNB antenna Gain	Omnidirectional antenna with 5 dBi gain			
UE noise figure	9 dB	9 dB	9 dB	9 dB
Average Spectral efficiency ($\overline{S_{eff}}$)	5.8 b/s/Hz	5 b/s/Hz	4.5 b/s/Hz	4.1 b/s/Hz
Link-level model to map SINR and bit rate	Model in Sec. 5.2.7 of [48] with maximum SINR= 30 dB and minimum SINR= -10 dB with alpha parameter=0.6			
N° of spectral efficiency samples for $f_{Seff}(s)$ generation	10^7 samples			
PRB Bandwidth (*B*)	360 kHz			

3.6.1.2. Results

In order to gain a first insight of the PRB estimation methodology behaviour, Fig. 25 and Fig. 26 show the required values of $N_{th,1}$ for Tenant 1 and $N_{th,2}$ for Tenant 2 when increasing the maximum number of GBR and non-GBR flows in the InH cell. Results show the implications of the SLA requirements in Table XI on the proposed methodology. On the one hand, it is observed that $N_{th,1}$ and $N_{th,2}$ experience a greater impact due to the increase of the number of GBR flows than due to the increase of non-GBR flows. The reason is that, since the average throughput requirement $\overline{Th^{min}_{nGBR,n}}$ for both tenants is low, just a small value of $N_{nGBR,n}$ in addition to the already computed $N_{GBR,n}$ is enough to fulfil the $\overline{Th^{min}_{nGBR,n}}$ requirement. On the other hand, it can be observed that the PRB requirement $N_{th,n}$ of Tenant 1 is higher than that of Tenant 2. For instance, for $u_{GBR,1}$=$u_{nGBR,1}$= $u_{GBR,2}$= $u_{nGBR,2}$=50 flows, Tenant 1 requires $N_{th,1}$=280 PRBs whereas Tenant 2 requires $N_{th,2}$=140 PRBs. This is due to two main factors: firstly, the GBR services of Tenant 1 have a higher GBR_n requirement than those of Tenant 2 and, secondly, the maximum degradation $P^{max}_{d,GBR,n}$ allowed to Tenant 1 is stricter than that of Tenant 2.

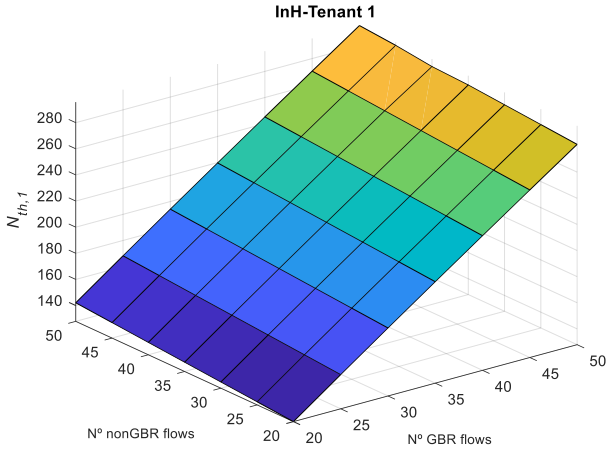


Fig. 25. Required resources for Tenant 1 $N_{th,1}$ for InH environment.

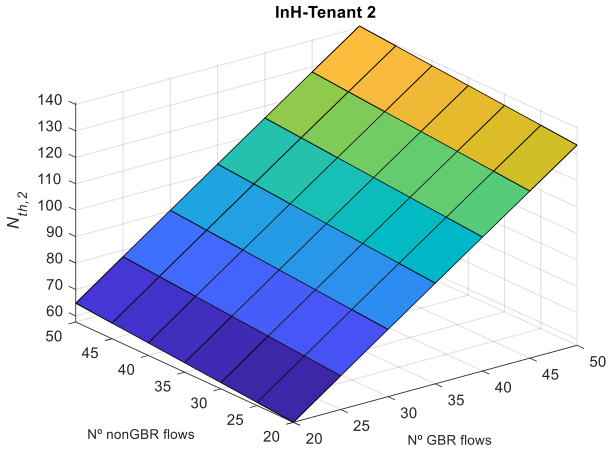


Fig. 26. Required resources for Tenant 2 $N_{th,2}$ for InH environment.

In order to gain further insight, the effect of the cell environment is analysed in Fig. 27, which includes the PRB requirements $N_{th,1}$ and $N_{th,2}$ for each of the tenants in the different cells of Table XII and in the selected

cells from the LTE dataset (i.e. cell#1, cell#2 and cell#3) for a fixed number of flows $u_{GBR,1}$=20, $u_{nGBR,1}$=20 for Tenant 1 and $u_{GBR,2}$=20, $u_{nGBR,2}$=20 for Tenant 2. In the figures, $N_{th,n}$ has been decoupled into the required PRBs due to GBR requirements $N_{GBR,n}$ and to non-GBR requirements $N_{nGBR,n}$. As previously observed, Tenant 1 has higher PRB requirement than Tenant 2 for all the selected cells due to its higher GBR_n and stricter $P^{max}_{d,GBR,n}$ values.

Furthermore, Fig. 27 reveals that the PRB requirement $N_{th,n}$ of a tenant can vary considerably among different cells. Differences are highly notable when comparing the results for the diverse simulation-based cells of Table XII. Taking as reference the InH cell, which is the cell with the best propagation conditions (i.e. higher S_{eff}) and, consequently, lower PRB requirement, significant differences of 50% and 55% are obtained in comparison with UMi and UMa cells, respectively. Differences grow until around 100% when comparing with the RMa cell, which is the one with the worst propagation conditions. As a matter of fact, PRB requirements of cells #1, #2 and #3 selected from the real LTE deployment dataset also present large differences among them, even though all could be assimilated to an UMa environment. For example, the PRB requirement in cells #1 and #3 only differs a 10%, but a difference of 30% is observed between cells #1 and #2.

The results obtained emphasise that the estimation of PRB requirements not only suffers variations when considering different environment cells, but also significant differences can take place between cells in the same environment. Consequently, the consideration of data analytics fed by CQI measurements extracted on a per cell basis allows an estimation of the required PRBs by a tenant tailored to the specificities of each cell.

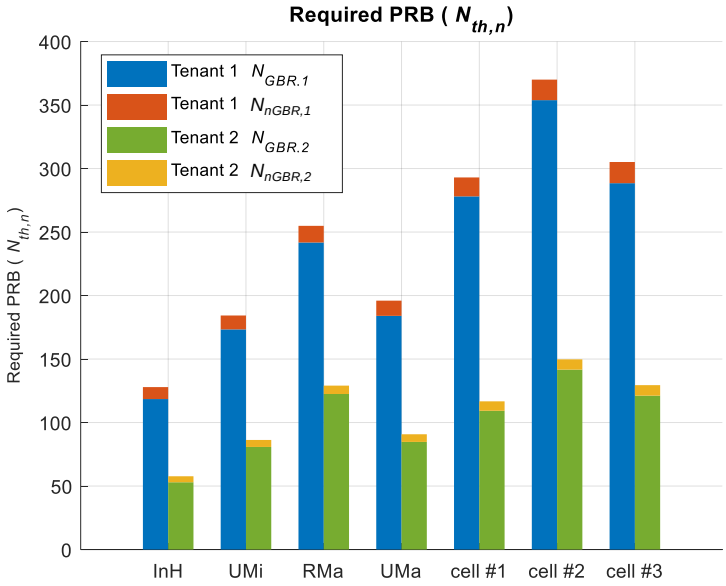


Fig. 27. $N_{GBR,n}$ and $N_{nGBR,n}$ required PRBs per cell and tenant.

3.6.2. Application to the solution for mission critical services

3.6.2.1. Considered scenario

The considered scenario under evaluation in this section has been defined under the scope of the 5G ESSENCE project, which considers a use case to address the support of public safety services. Specifically, the project defines a scenario that involves one public safety communications provider with a Mission Critical Push to Talk (MC PTT) service and one commercial communications provider (e.g. offering a video streaming service). The two service providers share the resources available at a deployed cellular infrastructure. The challenge consists on providing the required RAN slice in order to allocate the radio resources to the critical actors (e.g., the first responders), who inherently need services with prioritized and high quality requirements.

For the scenario configuration, 5G ESSENCE considers an incident in the central train station of a medium sized city. The public safety resources of the city involve three fire stations, three emergency services offices, three hospitals, the transport police that operates in the train station and the emergency coordination centre (112 headquarter). The communication between the participants in the incident is performed through the MC PTT service. In this context, the deployment of the cellular network includes 4 zones that divide the city into 4 coverage areas:

- Zone 1: includes the Fire Station 1, the Emergency Services Office 1 and the Hospital 1.
- Zone 2: includes the Fire Station 2, the Emergency Services Office 2 and the Hospital 2.
- Zone 3: includes the Fire Station 3, the Emergency Services Office 3, the Hospital 3, the Transport Police Headquarters and the Emergency Coordination.
- Zone 4: includes the Train station.

With this layout, the scenario is organised in three stages. Initially, only the basic personnel are deployed for surveillance operations. The occurrence of an incident triggers the transition to the second stage, attracting first responders to the scene and therefore, increasing communication requirements. Finally, the events lead to the declaration of an emergency state in the third stage, which involves another reconfiguration of the network to guarantee the communication between the PS forces. Table XIII summarizes the number of user connections (i.e., MC PTT calls) for Stages 1-3 as a result of the incident, which abruptly increase in zone 4 where the train station is located. For further details of the development of the incident and the calls between the different PS bodies, the reader is referred to [C4].

TABLE XIII. TOTAL NUMBER OF USER CONNECTIONS FOR EACH OF THE STAGES.

Stage	Zone 1	Zone 2	Zone 3	Zone 4
Stage 1 (Initial situation)	8	8	10	2
Stage 2 (The incident occurs)	12	8	7	30
Stage 3 (State of emergence)	12	13	7	60

Let us assume that the public safety communications provider of the MC PTT service makes use of a RAN Slice Instance (RSI) deployed on the shared cellular infrastructure that provides coverage in the different zones of the scenario. MC PTT service requirements are given in terms of the GBR_n value to be ensured to each MC PTT call and the maximum degradation probability $P^{max}_{d,GBR,n}$, which represents an important requirement for MC PTT services, due to its mission critical nature. Then, assuming that each one of the four zones of the scenario is covered by a different cell, the proposed methodology for the estimation of the required PRB resources is considered for dimensioning the PRB requirements of the MC PTT RSI. The considered configuration parameters of the cells are shown in Table XIV. The MC PTT service requirements are given by GBR_n=50 kb/s and different values of $P^{max}_{d,GBR,n}$ will be considered in the results.

TABLE XIV. CELL CONFIGURATION PARAMETERS.

Parameter	Value
Cell radius	288 m
gNB height	25 m
UE height	1.5 m
Minimum gNB-UE distance	35 m
Path Loss and Shadowing model	Urban Macrocell model of Sec. 7.4 of [45]
Shadowing standard deviation in LOS	4 dB
Shadowing standard deviation in NLOS	6 dB
Frequency	3.6 GHz
gNB transmitted power (dBm/PRB)	18 dBm
gNB antenna Gain	Omnidirectional antenna with 5 dBi gain
UE noise figure	9 dB
Average Spectral efficiency ($\overline{S_{eff}}$)	4.45 b/s/Hz
Link-level model to map SINR and bit rate	Model in Sec. 5.2.7 of [48] with maximum SINR= 30 dB and minimum SINR= -10 dB with alpha parameter=0.6
PRB Bandwidth (B)	180 kHz, corresponding to subcarrier spacing 15 kHz [43]
Number of PRBs per cell	106, corresponding to cell bandwidth 20 MHz [46]

3.6.2.2. Results

Fig. 28 plots the number of PRBs N_{th} required by the MC PTT RSI in each cell to support the traffic conditions existing in each stage of the scenario. The cell number corresponds to the zone number. Results are obtained for a maximum degradation probability $P^{max}_{d,GBR,n}$=0.1%. The figure shows that the impact of the traffic increase associated to the emergency incident is particularly relevant in the case of the cell 4 that provides service to the train station where the incident has occurred. Specifically, under normal conditions (stage 1) one single PRB is sufficient to serve the MC PTT communication needs. This PRB will be dynamically assigned by the scheduler to the different users on a short (i.e. 1 ms) so that each user gets on average the required GBR_n. In turn, with the traffic increase associated to the incident occurrence (stage 2) and the declaration of the emergency state (stage 3), this PRB becomes insufficient to serve all the users, thus requiring the extension of the MC PTT RSI capacity in the cell 4 with additional PRBs. Comparing the number of PRBs for this cell in Fig. 28 with the number of user connections indicated in Table XIII for

the different stages, it can be noticed that the number of required PRBs does not scale linearly with the number of users.

As for the rest of cells, even though the different stages of the incident involve some traffic increase, this can still be properly served in cells 1 and 3 with the same number of PRBs as during the normal operational conditions (stage 1), so RSI reconfiguration is not needed in these cells. For cell 2, an additional PRB is required in stage 3.

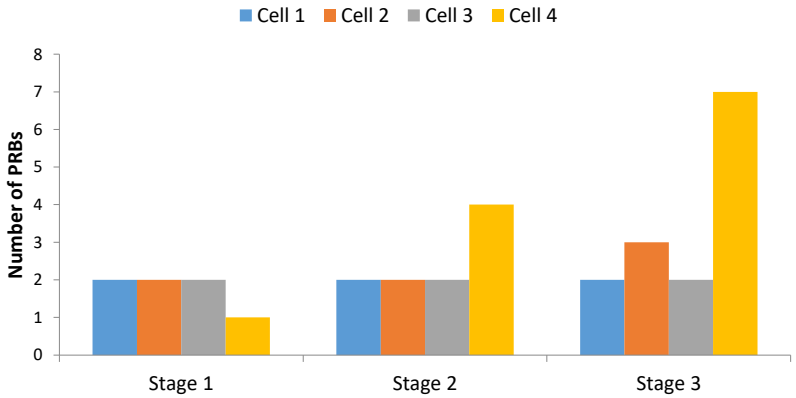


Fig. 28. Number of required PRBs for the MC PTT RSI in the different cells and stages for $P^{max}_{d,GBR,n}$=0.1%.

To analyse the impact of the service requirements, Fig. 29 plots the number of PRBs required by the MC PTT RSI in cell 4 for different values of the maximum degradation probability $P^{max}_{d,GBR,n}$. It is observed that the number of PRBs increases when reducing the value of $P^{max}_{d,GBR,n}$, which corresponds to more stringent requirements. In any case, the observed increase is quite moderate, as observed from the fact that in stages 2 and 3 a reduction of the degradation probability in two orders of magnitude from 1% down to 0.01% can be just achieved with two additional PRBs.

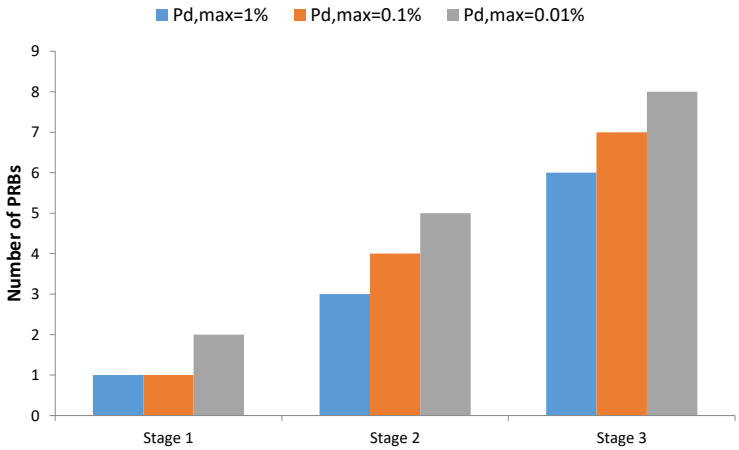


Fig. 29 Number of required PRBs for the MC PTT RSI in cell 4 for the different stages of the incident and with different values of $P^{max}_{d,GBR,n}$.

The impact of the increase of PRB requirements in the MC PTT RSI for the different stages of the incident over the performance obtained by the RSI of the commercial service provider is shown in Fig. 30. The figure plots the aggregate average throughput available in cell 4 for the different stages. The throughput is progressively reduced as more PRBs are allocated to the MC PTT RSI, since, for every reconfiguration of

this slice, the commercial RSI will have less PRBs available from the total in the cell. However, the reduction is relatively small, e.g. for the case $P^{max}_{d,GBR,n}$=0.01% the throughput in stage 3 is only 5.8% lower than the throughput in stage 1. This is due to the low GBR demanded by MC PTT calls, which allows supporting a large number of users with a relatively low number of PRBs.

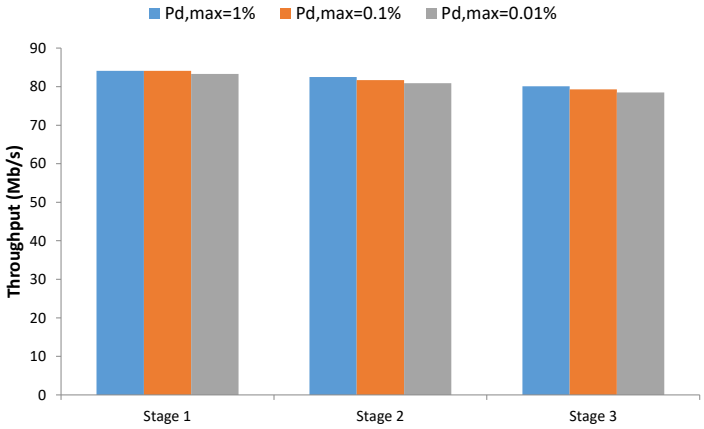


Fig. 30 Aggregate throughput available for the commercial RAN slice in cell 4 for different stages of the incident and different values of $P^{max}_{d,GBR,n}$.

3.7. Summary

This chapter has proposed an analytical methodology for estimating the required number of PRBs by a RAN slice for slice admission control decision making. To capture the cell specificities in the estimation, the characterisation of each cell in terms of spectral efficiency probability density function is derived by means of data analytics, after gathering and processing OAM cell performance measurements. Based on the obtained cell characterisation, the estimation of the required PRBs is performed on a per-cell basis in order to meet the SLA requirements of both GBR and non-GBR services.

The proposed methodology has been evaluated considering four different simulation-based cell environments (i.e. indoor hot-spot, urban micro-cell, urban macro-cell and rural-macro cell) and three cells selected from a real LTE deployment dataset. Results have revealed large differences in the estimated PRB requirements among different cell environments, presenting much lower PRB requirements the indoor hot-spot than the rural macro-cell. Also, significant differences can occur in cells with similar environments, particularly when data from real scenarios are considered. Therefore, the cell-tailored estimation of the required radio resources, as achieved with the proposed methodology fed by cell CQI measurements, is a must for proper SAC.

Moreover, the proposed PRB requirements estimation methodology has been applied to the dimensioning of the RAN slice instance that supports the MC PTT services. The analysed scenario, which has been defined in the 5G ESSENCE project, consists in an emergency scenario characterized by different stages that involve increasing communication needs associated to the occurrence of an incident and the eventual declaration of the emergency state. The results have shown that the traffic increase associated to the emergency occurrence involves reconfigurations in the MC PTT RAN slice to increase the number of allocated PRBs.

Chapter 4. Deep Reinforcement Learning-based framework for capacity sharing in RAN slicing scenarios

4.1. Introduction

In this section, a Deep Reinforcement Learning (DRL)-based solution to deal with the problem of capacity sharing in multi-tenant and multi-cell scenarios is proposed. The solution is able to dynamically provide the capacities to each tenant in the different cells in order to satisfy the fluctuating traffic demands of each tenant, satisfy the Service Level Agreement (SLA) established for each tenant and make an efficient use of the resources in each of the cells of the scenario. Specifically, the proposed solution is approach as a Multi-Agent Reinforcement Learning (MARL) solution that associates one Deep Q-Network (DQN)-based agent to each tenant, which learns the policy for jointly assigning the capacities to be provided to the tenant in the different cells of the scenario. The different DQN-based agents operate in a synchronous and cooperative manner since their decisions are performed at the same time and each agent is designed to find a solution that benefits jointly all the tenants. Furthermore, the solution has been formulated as a Self-Organising Network (SON) function, which is integrated in a RAN slicing management framework well aligned with the on-going 3GPP standardisation work on management and orchestration of network slicing.

The proposed solution has been evaluated considering diverse scenarios with different number of cells and traffic distributions among them. After illustrating the behaviour of the solution, the impact of the configuration parameters on the training and performance behaviour has been analysed. Also, the capability of generalization of the trained policies has been proved and illustrated when adding new tenants in a given scenario. Furthermore, the solution has been compared against the optimum solution obtained by an exhaustive search algorithm. Finally, the capability of the solution to deal with different levels of heterogeneity of the distribution of the traffic demands of the tenants among cells has been assessed.

The chapter has been organized as follows: Section 4.2 includes the literature review regarding capacity sharing solutions for RAN slicing scenarios. Section 4.3 describes the system model while Section 4.4 defines the MARL approach based on DQN. The definition of some performance metrics to analyse the model and the performance evaluation of the proposed solution follow in Section 4.5 and Section 4.6, respectively. Finally, Section 4.7 summarizes the main conclusions extracted from this chapter.

4.2. Literature review

The problem of capacity sharing in RAN slicing scenarios has been addressed by some prior works using different techniques and under different assumptions. The capacity sharing from a single cell perspective has been considered in [53]-[57]. Specifically, the problem is addressed in [53] by defining an exponential smoothing model, while in [54] it is formulated as an optimisation problem based on Karush Kuhn Tucker (KKT) conditions, and in [55] a biconvex problem is solved considering jointly the radio resources, caching and backhaul capacities in the RAN slicing process. Moreover, [56] and [57] establish the capacity provided to each tenant in a cell based on market-oriented models that aim at maximising the infrastructure provider's revenue. Other solutions address the problem of capacity sharing in multi-cell scenarios by using heuristic approaches [58]-[61]. In particular, capacity sharing is modelled in [58] as a winner bid problem solved by means of dynamic programming, while [59] uses an integer mathematical program and proposes a low complexity heuristic algorithm for associating resources to users. In turn, [60] proposes a fisher market game and [61] an iterative algorithm that adjusts the per-cell capacity provided to each tenant to be used for admission control.

Given the complexity of 5G networks and the inherent dynamic uncertainty of the wireless environment, Reinforcement Learning (RL) methods are potential candidates to deal with the capacity sharing problem, as they allow optimising dynamic decision-making problems in real time [62]. Among these methods, Deep Reinforcement Learning (DRL) approaches, which combine deep neural networks (DNNs) with RL, are particularly promising due to their capability to support large state and action spaces. The success of DRL started with the Deep Q-Network (DQN) [63] that combines DNN with Q-learning and since its launch successive extensions have been proposed such as Double DQN (DDQN) [64] and prioritized experience replay [65], among others, which mainly aim at enhancing the speed of learning and the stability of DQN but at the cost of increasing the complexity of the solution and the number of hyperparameters to configure [66]. Moreover, going beyond Q-learning, DNN have also been applied to other RL approaches, like the policy gradient-based methods used for continuous action spaces, such as the Deep Deterministic Policy Gradients (DDPG) algorithm [67], or actor-critic algorithms, such as the Asynchronous Advantage Actor Critic (A3C) algorithm [68].

DRL methods have already been used to approach the capacity sharing problem in multi-cell RAN slicing scenarios in some previous works [69]-[79]. In [69] and [70] the aggregated capacity reserved to each slice at network level is provided, respectively, by means of DQN and Deterministic Policy Gradients (DPG) combined with K-Nearest Neighbours (K-NN). A similar problem is addressed in [71] through Generative Adversarial Network (GAN)-DDQN. In turn, [72] and [73] provide the cell capacity share by first computing the aggregate capacity reserved to each slice at network level using a DQN agent associated to each slice and, then, applying a heuristic algorithm to obtain the cell capacity for each slice. Similarly, [74] provides the aggregate capacity allocated to each tenant at network level by means of a DQN agent and, then, the cell capacity for each tenant is computed by a knapsack algorithm. In [75], a DQN agent is used

to assign the cell capacity to all tenants in a single cell with virtualised Device to Device (D2D) communications. In [76] the RAN slice configuration and PRB allocation in a cell is performed by a two-level hierarchy scheme by using, respectively, DDPG and DDQN agents. The allocation of the cell capacity to all tenants in a single cell when considering DQN, DDQN and DDPG-based agents is assessed in [77]. Moreover, [78] and [79] propose an Ape-X-based approach to solve the capacity sharing problem, where the capacity provided to each slice in a single cell is provided by a different actor and all the actors base their policy on a common learner.

While the DRL-based capacity sharing approaches in [69]-[71] and [75]-[77] propose a single-agent to provide the joint solution for all the tenants, the approaches in [72]-[74], [78] and [79] tackle the problem as a Multi-Agent Reinforcement Learning (MARL) approach [80]. In MARL, multiple agents interact with a common environment in order to learn each agent's policy according to RL methods. The use of MARL to tackle the capacity sharing problem by associating one agent to each tenant exhibits relevant advantages with respect to a single-agent approach that allocates the capacity to all the tenants. On the one hand, it is more scalable as it allows easily adding/removing new tenants in the scenario simply by adding/removing the corresponding agent and without modifying the structure of the DNN, which in the single-agent case would depend on the number of tenants. On the other hand, it increases the speed of learning, since the dimensions of the state-action spaces are more reduced as the agent only needs to account for the states and actions related to the tenant and the training of the different agents can be performed independently of the others. The benefits of MARL have been exploited in [72]-[74], where each agent determines the aggregated assigned capacity over all the cells to a tenant, as well as in [78] and [79], where each agent determines the capacity assigned to a tenant in a single cell. However, none of the previous works have proposed a MARL approach where an agent is able to directly provide the capacities assigned to its associated tenant in the different cells in a multi-cell scenario. This is relevant for the management of RAN slices in multi-cell scenarios from a system-level perspective because, on the one hand, Service Level Agreements (SLA) are defined for geographical areas covering multiple cells and, on the other hand, the traffic of one tenant across the different cells may exhibit time and space heterogeneities. Therefore, it is important that an agent learns how to make joint decisions for multiple cells and the proposed MARL approach intends to fill this gap. Besides, another gap to be filled is that the previous MARL-based approaches in [72]- [74] provide limited insights about multi-agent design features that are relevant from a practical perspective, such as the interaction between the different agents or the training-related aspects of the multi-agent approach. Only the work in [78] and [79] provides some details in this respect for the Ape-X solution, but being restricted to the operation on a single cell basis.

The main contributions of the solution proposed in this chapterand its novelties with respect to previous works are the following:

- In the proposed MARL approach, a DQN agent learns the policy for jointly assigning the capacities to be provided to a tenant in the different cells of the scenario. This is a difference with respect to

prior multi-agent approaches in which an agent either assigns the capacity for a single cell, [78] and [79], or assigns the aggregate capacity for all cells, [72]-[74]. Furthermore, a synchronous and cooperative operation of the different agents is considered in the proposed solution since their decisions are performed at the same time and each agent is designed to find a solution that benefits jointly all the tenants. This also constitutes a relevant difference with respect to other approaches, such as [72] and [73].

- The proposed MARL approach addresses the capacity sharing when considering the SLA for each tenant as an aggregate across the multiple cells in the scenario in order to capture the total amount of capacity to be provided to each tenant. Instead, other approaches such as [69]- [79] just consider the SLA specified in terms of the QoS parameters defined at user level, but without enforcing any aggregate capacity per tenant. In this way, the definition of the state in the MARL solution proposed here includes the parameters of the SLA, which allows the agents to adapt to different SLA requirements without the need of performing a new training when the SLA values change, as required in [72]-[75].

- A key feature of the proposed solution is that the policy learning process can be conducted by a single agent and then the learnt policy can be generalised and directly applied by the other agents, thus reducing the complexity of training in multi-agent scenarios. This leads to a scalable solution where e.g., the addition of a new tenant does not require any re-training. To the author best knowledge, little effort has been conducted by previous works on capacity sharing to address these training-related aspects in detail. In this respect, [78] and [79] use Ape-X for learning a single policy that jointly uses the experiences of multiple tenants. Instead, the general policy considered in the present work can be learnt based on the experiences of a single tenant, thus simplifying the learning process.

- The practicality of the proposed approach is enforced with respect to previous capacity sharing solutions by formulating the solution as a Self-Organising Network (SON) function, which is integrated in a RAN slicing management framework well aligned with the on-going 3GPP standardisation work on management and orchestration of network slicing [52], [81], [82].

4.3. System model

Let us consider an Infrastructure Provider (InP) that owns a NG-RAN infrastructure, which is composed of N cells with diverse deployment characteristics (i.e., cell radius, transmission power, frequency of operation). Assuming 5G New Radio (NR) technology, each cell n has a total of W_n Physical Resource Blocks (PRBs) with a PRB bandwidth B_n, which provide a total cell capacity c_n (b/s), defined as $c_n = W_n \cdot B_n \cdot S_n$, where S_n is the average spectral efficiency at cell n. Then, the total system capacity C is obtained by aggregating c_n for all the cells n=1…N. The InP shares its NG-RAN infrastructure among K

tenants by providing each tenant k with a RAN Slice Instance (RSI), which corresponds to the Network Slice Subnet Instance (NSSI) for the RAN part in the 3GPP terminology.

In order to satisfy the service requirements, a SLA is established between the InP and each of the tenants. Based on this SLA, the following requirements are established for the k-th tenant:

- Scenario Aggregated Guaranteed Bit Rate ($SAGBR_k$): the aggregated capacity to be provided across all cells to tenant k, if requested.
- Maximum Cell Bit Rate ($MCBR_{k,n}$): Maximum bit rate that can be provided to tenant k in cell n. This parameter is defined by the InP to avoid that a single tenant uses all the capacity in a cell under highly extreme heterogeneous spatial load distributions of tenants demanding excessive capacity in certain cells.

These requirements are considered in the creation process of each RSI and its fulfilment is responsibility of the InP. For this purpose, it relies on the RSI Lifecycle Management (LCM) for creating, modifying, optimising and terminating RSIs. According to the 3GPP management model [52], the RSI LCM falls under the scope of the Network Slice Subnet Management Function (NSSMF) for the RAN part of a network slice, denoted hereafter as RAN NSSMF.

The RAN NSSMF is in charge of the correct operation and dynamic configuration of the RSIs. In particular, the proposed capacity sharing function is considered as a part of the RAN NSSMF to optimise the amount of radio resources (and associated capacity) to be provided to each RSI, which is a challenging task. This provided capacity needs to be dynamically updated depending on the RSI traffic demands, which are not homogeneous among the different cells in the NG-RAN and they can fluctuate over the time. Therefore, the capacity sharing solution considered here is designed as a SON function that automatically and dynamically adjusts the capacity provided to each RAN slice across the different cells. Indeed, this vision is aligned with the recent study conducted at 3GPP on the SON functionalities for 5G [81]. This study identifies the so-called cross-slice network resource optimization use case intended to optimise the allocation of physical and virtual resources across multiple network slice instances. Therefore, the proposed capacity sharing solution can be regarded as a specific solution to this use case for the RAN.

Specifically, the proposed capacity sharing SON function dynamically tunes the capacity share for each tenant in time steps of duration Δt (in the order of minutes) in order to adapt to the spatial and temporal traffic variations among the different cells, minimise SLA breaches (i.e., violations) in the system and optimise the resource utilisation of the different cells in the system. The capacity share $\boldsymbol{\sigma_k(t)}$ of tenant k in time step t is defined as $\boldsymbol{\sigma_k(t)}=[\ \sigma_{k,1}(t),\ldots,\ \sigma_{k,n}(t),\ \ldots,\ \sigma_{k,N}(t)]$, where each component $\sigma_{k,n}(t)$ corresponds to the proportion of the total capacity (i.e. proportion of the total PRBs W_n) in cell n provided to that tenant during time step t and ranges $0 \leq \sigma_{k,n}(t) \leq MCBR_{k,n}/c_n$. Note that the capacity share solution in a cell cannot exceed the total capacity of the cell, so that $\sum_{k=1}^{K} \sigma_{k,n}(t) \leq 1$. The capacity share $\boldsymbol{\sigma_k(t)}$ needs to be upgraded periodically on a per-minutes basis in order to adapt to the traffic demands. Following the current 5G

Network Resource Model (NRM) defined by 3GPP [52] the capacity share can be configured on a cell using the *RRMPolicyRatio* attribute, which specifies the percentage of radio resources (e.g., PRBs) to be allocated on a per-slice basis, which is further discussed in Section 5.2. The basic notations of the proposed capacity sharing SON function are summarized in Table XV.

TABLE XV. LIST OF ABBREVIATIONS AND NOTATIONS.

Notation	Definition
N	Total number of cells.
W_n	Total number of PRBs in cell n.
B_n	PRBs' bandwidth in cell n.
c_n	Total capacity in cell n.
S_n	Average spectral efficiency in cell n.
C	System capacity.
K	Total number of tenants.
$SAGBR_k$	Scenario Aggregated Guaranteed Bit Rate to tenant k.
$MCBR_{k,n}$	Maximum Cell Bit Rate to tenant k in cell n.
Δt	Duration of a time step.
$\boldsymbol{\sigma_k(t)}$	Capacity share of tenant k in time step t.
$\sigma_{k,n}(t)$	Proportion of c_n of cell n provided to tenant k during time step t.
π_k	Policy learnt by the agent associated to tenant k.
$o_{k,n}(t)$	Offered load of tenant k in cell n.
$\rho_{k,n}(t)$	Resource usage of tenant k in cell n.
$T_k(t)$	Aggregated throughput of tenant k across all cells.
$\boldsymbol{s_k(t)}$	State of tenant k in time step t.
$s_{k,n}(t)$	State of tenant k in cell n in time step t.
$\boldsymbol{a_k(t)}$	Action selected for tenant k in time step t.
$a_{k,n}(t)$	Action selected for tenant k in cell n in time step t.
Δ	Action increase step.
$r_k(t)$	Reward obtained by tenant k in time step t.
$\rho_n^A(t)$	Fraction of available PRBs not used by any tenant in the cell.
$\sigma_n^A(t)$	Available capacity share in cell n not assigned to any tenant.
$\delta_k^{(1)}(t)$, $\delta_k^{(2)}(t)$	Reward factors.
φ_1, φ_2	Reward factors' weights.
$O_k(t)$	Aggregated offered load of tenant k across all cells.
$O(t)$	Aggregated offered load of all tenants in all cells.
$\beta_k(t)$	Guaranteed capacity not required by other tenants at time step t.
γ	Discount factor.
$Q_k(\boldsymbol{s_k}, \boldsymbol{a_k}, \theta_k)$	Q-network associated to tenant k with weights θ_k.
D_k^l	Experience dataset of agent associated to tenant k with length l.
$U(D_k^l)$	Mini-batch of experiences.
π_k^ε	ε-Greedy policy with probability of selecting a random action ε.
$L(\theta_k)$	Average mean squared error loss.
τ	Learning rate.
$e_{k,j}$	Experience j of tenant k.
$A_k(t)$	Assigned capacity to tenant k at time step t.
R_k	Average reward of tenant k.
SS_k	Average SLA satisfaction of tenant k.
U	Average system utilization.

4.4. Multi-agent reinforcement learning approach for capacity sharing

The capacity sharing SON function has been addressed as a MARL approach in order to deal with the complexity of the computation of $\boldsymbol{\sigma_k(t)}$ in multi-cell scenarios. In the proposed approach, each RL agent is associated to a tenant k in the system and centrally learns the policy π_k to tune $\boldsymbol{\sigma_k(t)}$ dynamically by interacting with the environment. The selected RL method for deriving the policy π_k at the k-th agent is the DQN algorithm due to three main reasons. First, DQN has been designed to support high dimension state

and action spaces, which is achieved with the use of DNN. This is convenient for the capacity sharing problem since the consideration of multiple cells and the randomness in the traffic demands in each of the cells can result in large state and action spaces. Second, the learning in DQN is performed by bootstrapping, i.e., the policy is progressively updated by considering single samples of experience instead of considering all the samples until reaching a certain goal or finishing an episode, like in Monte Carlo simulations [83]. This is suitable for the case of capacity sharing since a continuous learning of π_k is desired, rather than in episodes. Third, in relation to other DRL approaches such as DDQN or DDPG, preliminary results in a previous study [77] showed that, despite having differences in terms of the practicality of the implementation (e.g., speed of the training process, number of hyperparameters to configure, etc.), their performance in a single-agent and single-cell scenario was very similar to that of DQN. In this respect, DQN was considered as a good design choice, considering the trade-off between practicality and achieved performance.

The scheme of the proposed solution is shown in Fig. 31. The proposed capacity sharing SON function falls within the scope of the RAN NSSMF of the 3GPP management system. The function is composed of K DQN agents, each one associated to one tenant. The InP provides as input the service profile parameters associated to each tenant, which include the SLA parameters. Moreover, the SON function includes a monitoring module, which collects performance measurements of the tenants in the different cells and provides them to the processing module. The performance measurements of each tenant can be obtained in each time step e.g., using the NSSI performance data file reporting service defined in [82] through Performance Management (PM) services interface. The performance measurements considered here for tenant k include:

- Offered load ($o_{k,n}(t)$): Requested capacity by the tenant in the n-th cell during the last time step, i.e. period (t-Δt,t]. This parameter is obtained by aggregating the capacity requirement by all the users in cell n that belong to tenant k.

- Resource usage ($\rho_{k,n}(t)$): Fraction of PRBs occupied by the k-th tenant in the n-th cell during the last time step. It is computed as $\min(\sigma_{k,n}(t), o_{k,n}(t)/c_n)$.

- Throughput ($T_k(t)$): Aggregated throughput experienced by tenant k across all cells during the last time step. This is obtained by aggregating the throughput in each cell n given by $\min(o_{k,n}(t), \sigma_{k,n}(t)\cdot c_n)$.

Furthermore, a pre-processing module is also proposed to be included within the SON function, which computes the inputs to each DQN agent based on the performance measurements provided by the monitoring module and on the allocated capacity share $\boldsymbol{\sigma_k(t\text{-}1)}$ during the last time step t-1.

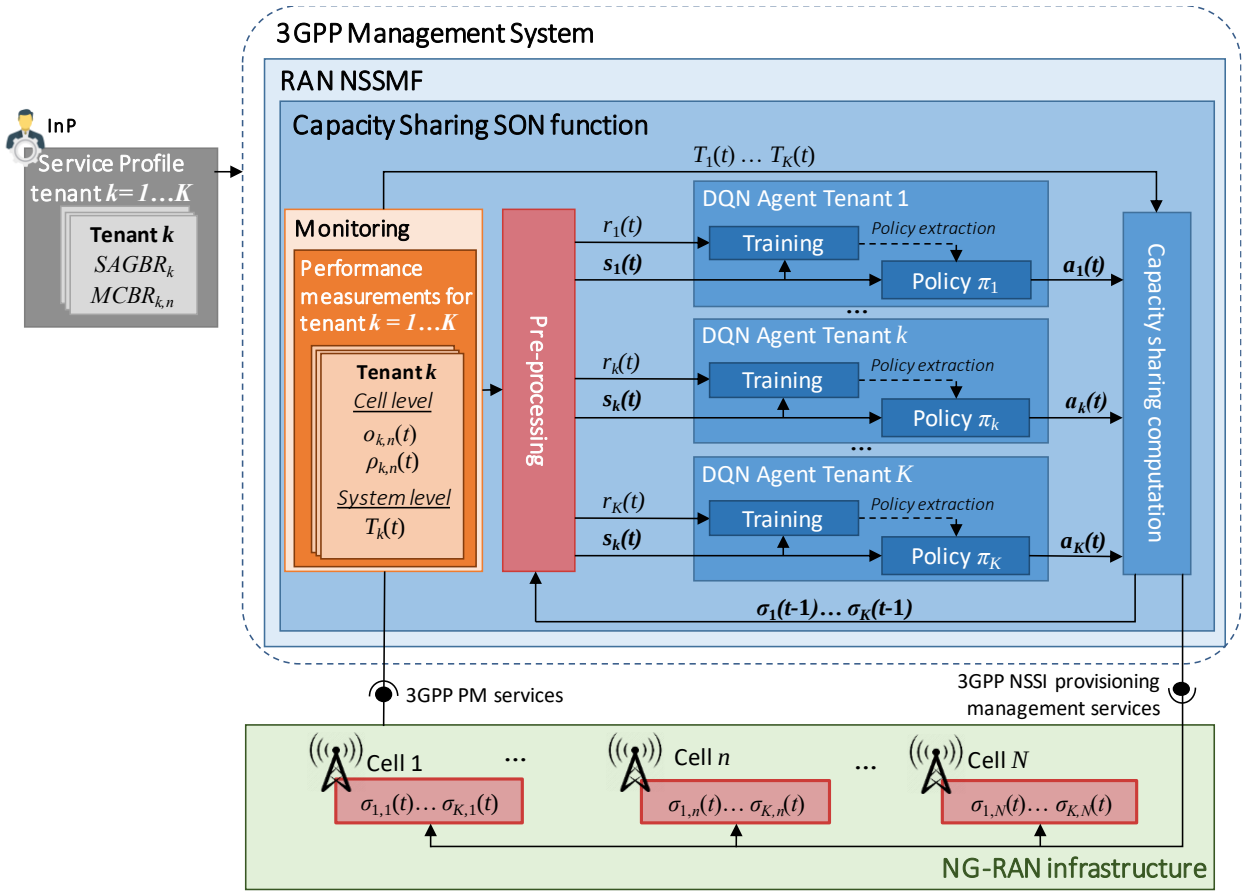


Fig. 31. MARL solution scheme.

Then, at each time step *t*, the DQN agent associated to tenant *k* obtains the state $\boldsymbol{s_k(t)}$ of the environment. Based on $\boldsymbol{s_k(t)}$, the agent selects an action $\boldsymbol{a_k(t)}$ according to the trained policy π_k by the DQN agent. Once all the actions for all the agents have been obtained, the capacity sharing computation module computes the resulting $\boldsymbol{\sigma_k(t)}$ for all the tenants, avoiding unfeasible $\boldsymbol{\sigma_k(t)}$ solutions (e.g., allocation of more capacity than available in a cell). Next, the computed capacity sharing values $\boldsymbol{\sigma_k(t)}$ are communicated to the different cells in the NG-RAN environment through 3GPP interfaces for NSSI provisioning management services [16]. Moreover, the *k*-th agent is also provided with the reward signal $r_k(t)$ as a result of the last performed action $\boldsymbol{a_k(t\text{-}1)}$, which jointly with the $\boldsymbol{s_k(t)}$ are used for training the DQN agent associated to tenant *k*. The specific definitions of the state, action and reward signals in the proposed DQN agents and the operation of the capacity sharing computation module are described in the following.

4.4.1. State

The state obtained by the *k*-th tenant's agent at time *t* from the network environment is denoted as $\boldsymbol{s_k(t)} = [s_{k,1}(t), \ldots, s_{k,n}(t), \ldots, s_{k,N}(t), SAGBR_k/C, \sum_{k'=1,k'\neq k}^{K} SAGBR_{k'}/C]$, where each component $s_{k,n}(t)$ corresponds to the state of the tenant in cell *n*, given by $< \rho_{k,n}(t), \rho_n^A(t), \sigma_{k,n}(t\text{-}1), \sigma_n^A(t\text{-}1), MCBR_{k,n}/c_n >$. The component $\rho_n^A(t)$ is the fraction of available PRBs not used by any tenant in the cell, that is:

$$\rho_n^A(t)=1-\sum_{k=1}^{K}\rho_{k,n}(t) \tag{46}$$

Similarly, $\sigma_n^A(t)$ is the available capacity share in cell n not assigned to any tenant, given by:

$$\sigma_n^A(t)=1-\sum_{k=1}^{K}\sigma_{k,n}(t) \tag{47}$$

4.4.2. Action

At time t, the k-th tenant's agent selects the joint action $\boldsymbol{a_k(t)}= [a_{k,1}(t), ..., a_{k,n}(t), ..., a_{k,N}(t)]$, which is composed of the cell-specific actions $a_{k,n}(t)$ for each cell n=1...N. The cell-specific action determines the variation in the capacity share $\sigma_{k,n}(t)$ to be applied in the following time step t in cell n. In order to achieve a gradual tuning of the capacity share, the cell-specific action can take three possible values $a_{k,n}(t)\in\{\Delta,0,-\Delta\}$, corresponding to increasing the capacity share in a step of Δ, maintaining it or decreasing it in a step of Δ. Consequently, the action space for $\boldsymbol{a_k(t)}$ corresponds to all the possible combination vectors of the three possible action values for each of the cells, which results in 3^N possible actions for each tenant. It is worth noting that other approaches in which the cell-specific action could take more than three possible values could also be feasible. However, they would imply a much larger action space, and thus, a longer process for learning the policy π_k. The actions $\boldsymbol{a_k(t)}$ are provided to the capacity sharing computation module to obtain the resulting capacity share solution $\boldsymbol{\sigma_k(t)}$, as detailed in Section 4.4.5.

4.4.3. Reward

In order to assess to what extent the last action $\boldsymbol{a_k(t\text{-}1)}$ was adequate for the previous state $\boldsymbol{s_k(t\text{-}1)}$, a reward $r_k(t)$ is provided to the k-th tenant agent, which is defined as:

$$r_k(t)=\delta_k^{(1)}(t)^{\varphi_1}\cdot\delta_k^{(2)}(t)^{\varphi_2} \tag{48}$$

which considers two main factors, $\delta_k^{(1)}(t)$ and $\delta_k^{(2)}(t)$, defined in the following, and their corresponding weights, φ_1 and φ_2. These weights φ_1 and φ_2 are assumed to be defined by the InP according to its own criteria in establishing the importance of the factors $\delta_k^{(1)}(t)$ and $\delta_k^{(2)}(t)$.

The first factor, denoted as $\delta_k^{(1)}(t)$, intends to promote the satisfaction of the SLA of tenant k and the provisioning of enough capacity to satisfy its offered load. It is defined as:

$$\delta_k^{(1)}(t)=\begin{cases}\dfrac{T_k(t)}{O_k(t)} & \text{if } O(t)\leq C\\ \min\left(\dfrac{T_k(t)}{\min(SAGBR_k+\beta_k(t),O_k(t))},1\right) & \text{Otherwise}\end{cases} \tag{49}$$

where $O_k(t)$ corresponds to the aggregated offered load of tenant k among all the cells at time step t, bounded by the $MCBR_{k,n}$ of each of the cells, given by:

$$O_k(t)=\sum_{n=1}^{N}\min(o_{k,n}(t),MCBR_{k,n}) \quad (50)$$

Moreover, $O(t)$ in (49) corresponds to the aggregated offered load $O_k(t)$ of all tenants at time step t while $\beta_k(t)$ captures the guaranteed capacity not required by other tenants at time step t, given by:

$$\beta_k(t)=\sum_{\substack{k'=1\\k'\neq k}}^{K}\max(SAGBR_{k'}-O_{k'}(t),0) \quad (51)$$

which reaches $\beta_k(t)=0$ when the offered load $O_k(t)$ of all the tenants is higher or equal than their $SAGBR_k$. Based on these parameters, the definition of $\delta_k^{(1)}(t)$ in (49) considers two different situations. The first condition in (49) considers the situation when there is enough capacity to satisfy the offered load of all the tenants $O(t)$ and, hence, $\delta_k^{(1)}(t)$ is given by the ratio between the throughput obtained by tenant k and its offered load, being maximum, i.e., $\delta_k^{(1)}(t)=1$, when the throughput equals the offered load. In turn, the second condition in (49) considers the situation when the total system capacity C is not enough to satisfy the offered load requirement of all the tenants $O(t)$. In this situation, $\delta_k^{(1)}(t)$ is given by the ratio between the throughput of tenant k and the minimum between its offered load and its $SAGBR_k$ increased by the unused capacity left by the other tenants $\beta_k(t)$. Then, in this case $\delta_k^{(1)}(t)$ will be maximum (i.e., $\delta_k^{(1)}(t)=1$) when the throughput equals the offered load $O_k(t)$ or, in the case that the offered load is higher than $SAGBR_k+\beta_k(t)$ when at least the throughput ensures this later value.

The fact that the definition of $\delta_k^{(1)}(t)$ depends on the aggregated offered load of all tenants $O(t)$ contributes to a collaborative behaviour between tenants, as in situations of overload this factor will promote actions that allow assuring the SLA of all tenants, avoiding those actions that would only benefit the k-th tenant at the cost of degrading the performance of the rest.

The second factor in the reward, $\delta_k^{(2)}(t)$, aims at measuring the degree of capacity overprovisioning and is defined by the ratio between the system throughput provided to the k-th tenant $T_k(t)$ and its provided capacity during the last time step, that is:

$$\delta_k^{(2)}(t)=\frac{T_k(t)}{\sum_{n=1}^{N}c_n\cdot\sigma_{k,n}(t\text{-}1)} \quad (52)$$

4.4.4. DQN Agent

The DQN agent of tenant k centrally learns the policy π_k that determines the actions to be executed in each cell. The proposed DQN agent executes the DQN algorithm of [63] but particularised to the state, action and reward signals previously introduced.

The objective of DQN, as a value-based RL algorithm, is to find the optimal policy π_k^* that maximises the discounted cumulative future reward $d_k(t)$, computed as:

$$d_k(t)=\sum_{j=0}^{\infty}\gamma^j r_k(t+j+1) \tag{53}$$

where γ is the discount factor, which ranges $0\leq\gamma\leq1$, used to place more emphasis on immediate rewards. Finding π_k* can be performed by obtaining the optimal action-value function $Q_k^*(\boldsymbol{s_k},\boldsymbol{a_k})$, which is the maximum expected discounted cumulative reward starting at time step t from $\boldsymbol{s_k}$, taking the action $\boldsymbol{a_k}$ and following the policy π_k:

$$Q_k^*(\boldsymbol{s_k},\boldsymbol{a_k})=\max_{\pi_k} E\left[d_k(t)\middle|\begin{matrix}\boldsymbol{s_k(t)}=\boldsymbol{s_k}\\ \boldsymbol{a_k(t)}=\boldsymbol{a_k}\\ \pi_k\end{matrix}\right] \tag{54}$$

This last expression can be decomposed into the Bellman Equation, which allows expressing Q_k*($\boldsymbol{s_k},\boldsymbol{a_k}$) in a recursive form:

$$Q_k^*(\boldsymbol{s_k},\boldsymbol{a_k})=E\left[r_k(t+1)+\gamma\max_{\boldsymbol{a_k'}}Q_k^*(\boldsymbol{s_k(t+1)},\boldsymbol{a_k'})\middle|\begin{matrix}\boldsymbol{s_k(t)}=\boldsymbol{s_k}\\ \boldsymbol{a_k(t)}=\boldsymbol{a_k}\end{matrix}\right] \tag{55}$$

According to $Q_k^*(\boldsymbol{s_k},\boldsymbol{a_k})$, the optimal policy can be found by selecting $\boldsymbol{a(k)}$ greedily for each state $\boldsymbol{s_k}$ that is:

$$\pi_k^*=\underset{\boldsymbol{a_k}}{argmax}\,Q_k^*(\boldsymbol{s_k},\boldsymbol{a_k}) \tag{56}$$

Generally, RL algorithms approximate the optimal $Q_k^*(\boldsymbol{s_k},\boldsymbol{a_k})$ by updating the approximated function $Q_k(\boldsymbol{s_k},\boldsymbol{a_k})$ iteratively based on the time difference (TD) error at each time t. The TD error is defined as the difference between $r_k(t+1)+\gamma\max_{\boldsymbol{a_k'}}Q_k^*(\boldsymbol{s_k(t+1)},\boldsymbol{a_k'})$, denoted as TD target, and the approximated value $Q_k(\boldsymbol{s_k},\boldsymbol{a_k})$. In the case of DQN, a non-linear approximation of $Q_k^*(\boldsymbol{s_k},\boldsymbol{a_k})$ is performed by using a DNN, denoted as Q-network, with weights θ_k, so that $Q_k^*(\boldsymbol{s_k},\boldsymbol{a_k})\approx Q_k(\boldsymbol{s_k},\boldsymbol{a_k},\theta_k)$**.** The inputs of the Q-network are the different components of the state $\boldsymbol{s_k}$, while the outputs correspond to the values of $Q_k(\boldsymbol{s_k},\boldsymbol{a_k},\theta_k)$ for each possible action $\boldsymbol{a_k}$. Therefore, the policy π_k selects the action that maximises the output of the Q-network as:

$$\pi_k=\underset{\boldsymbol{a_k}}{argmax}\,Q_k(\boldsymbol{s_k},\boldsymbol{a_k},\theta_k) \tag{57}$$

The use of a DNN to approximate $Q_k^*(\boldsymbol{s_k},\boldsymbol{a_k})$ allows dealing with large state and action spaces. However, the use of non-linear approximation functions such as DNNs can imply instabilities or even divergence in the learning process due to: (i) correlations in sequential observations; (ii) correlations between action-values $Q_k(\boldsymbol{s_k},\boldsymbol{a_k},\theta_k)$ and the TD-target; (iii) the fact that small updates of $Q_k(\boldsymbol{s_k},\boldsymbol{a_k},\theta_k)$ may change the policy, dramatically, which can lead to changes of the distribution of the data collected from the environment. In order to avoid these effects, a DQN agent is composed of different elements, listed in the following:

- **Evaluation DNN ($Q_k(\boldsymbol{s_k},\boldsymbol{a_k},\theta_k)$)**: corresponds to the main approximation function of the expected reward function $Q_k(\boldsymbol{s_k},\boldsymbol{a_k})$. This function is trained off-line and is used to extract the policy π_k to

select the actions to be performed in the environment. When starting the learning process, the weights θ_k are initialised randomly.

- **Target DNN ($Q_k(\boldsymbol{s_k},\boldsymbol{a_k},\theta_k^-)$):** this is another Q-network with the same structure as the evaluation DNN but with weights θ_k^-. It is used to obtain the TD-Target as $r_k(t)+\gamma \max_{\boldsymbol{a_k'}} Q_k(\boldsymbol{s_k(t)},\boldsymbol{a_k'},\theta_k^-)$. Instead of updating the weights θ_k^- every time step, they are updated every M time steps with the weights of the evaluation DNN $\theta_k^- = \theta_k$. Consequently, the computation of the TD error, which depends on the target DNN, is no longer dependant on rapidly fluctuating estimates of the Q-values, as the target DNN only changes every M time steps but remains fixed the rest of the time.

- **Experience Dataset (D_k^l):** a dataset D_k^l of length l is used to store the experiences of each agent. The stored experience for the agent associated to tenant k at time step t is represented by the experience tuple $<\boldsymbol{s_k(t\text{-}1)}, \boldsymbol{a_k(t\text{-}1)}, r_k(t), \boldsymbol{s_k(t)}>$. The use of the dataset allows randomly selecting mini-batches of experiences $U(D_k^l)$ with length J to update of the weights θ_k of the evaluation DNN. The use of D_k^l and the random minibatches for the learning of $Q_k(\boldsymbol{s_k},\boldsymbol{a_k},\theta_k)$ is called *Experience Replay* and has several benefits. First, randomly selecting experiences from the dataset D_k^l breaks the temporal correlations in the training data, which may lead to an inefficient learning. Second, past experiences can be reused, allowing for a greater data efficiency. Third, when not using experience replay and updating $Q_k(\boldsymbol{s_k},\boldsymbol{a_k},\theta_k)$ on-policy, i.e. based on real time experiences, the current values of $Q_k(\boldsymbol{s_k},\boldsymbol{a_k},\theta_k)$ determine the action that will lead to the next state, which will be used to update θ_k and will determine all the future experiences. This may lead to unwanted feedback loops where the values $Q_k(\boldsymbol{s_k},\boldsymbol{a_k},\theta_k)$ can get stuck in poor local minimum or diverge. This effect is smoothed when using experience replay, since the data used for training is averaged over many of the previous states.

The training operation of the different DQN agents in the MARL approach has been summarized in *Algorithm* 2. The process of training of each DQN agent can be split into two main processes: the data collection (lines 4-13 of *Algorithm* 2) and the update of the weights θ_k of the evaluation DNN (lines 14-22 of *Algorithm* 2).

For the DQN agent associated to tenant k, the process of data collection consists in gathering experiences from the network environment and storing them in the experience dataset D_k^l, which is performed in time steps of Δt. For each time step t, the DQN agent observes the state of the environment $\boldsymbol{s_k(t)}$ and, accordingly, triggers an action $\boldsymbol{a_k(t)}$ based on an ε-Greedy policy π_k^ε that chooses actions according to (57) with probability 1- ε and a random action with probability ε. By using the ε-Greedy policy π_k^ε, the agent can explore new states that would not be visited, which improves the learning behaviour of the policy π_k. Moreover, the reward $r_k(t)$ is obtained, which assesses the suitability of the last performed action $\boldsymbol{a_k(t\text{-}1)}$ for the last state $\boldsymbol{s_k(t\text{-}1)}$. Then, the experience $< \boldsymbol{s_k(t\text{-}1)}, \boldsymbol{a_k(t\text{-}1)}, r_k(t), \boldsymbol{s_k(t)} >$ is stored in the dataset D_k^l. When

the dataset D_k^l is full (i.e., l experiences are stored), old experiences are removed from the dataset to save new ones. Note that during the initial steps of the data collection, the actions are selected completely randomly (i.e., ε=1 is considered in the π_k^ε) from the environment in order to explore several states and start filling the dataset D_k^l with experiences.

Algorithm 2. MARL DQN training

Initialize DNN counter m=0.
For t=0… MaxNumberOfTrainingSteps
  For k=0…K
    Collect global state $\boldsymbol{s_k(t)}$ by obtaining $s_{k,n}(t)$ for $n=1…N$ cells.
    Generate random ε' (ε'=1 for the initial steps).
    If $\varepsilon'<\varepsilon$
      Choose randomly action $\boldsymbol{a_k(t)}$.
    Else
      Obtain action according to π_k in (57).
    End if
    Obtain reward $r_k(t)$ as a result of last action $\boldsymbol{a_k(t-1)}$.
    If $D_l(k)$ is full (l samples are stored), remove the oldest one.
    Store experience < $\boldsymbol{s_k(t-1)}$, $\boldsymbol{a_k(t-1)}$, $r_k(t)$, $\boldsymbol{s_k(t)}$ > in D_k^l.
    Randomly sample a minibatch of experiences $U(D_k^l)$ from D_k^l of length J.
    Compute the loss function $L(\theta_k)$ by (58).
    Compute the mini-batch gradient descent $\nabla L(\theta_k)$ by (59).
    Update weights θ_k of evaluation DNN by (60).
    If m==M
      Update the weights of target DNN $\theta_k^- = \theta_k$ and set m=0.
    Else
      m=m+1
    End if
  End for
  Compute $\boldsymbol{\sigma_k(t)}$ for k=1…K by applying *Algorithm* 3 for all cells.
End for

The process of updating the weights θ_k of the evaluation DNN is based on the experiences stored in the experience dataset. For each update of $Q_k(\boldsymbol{s_k},\boldsymbol{a_k},\theta_k)$, a minibatch of J experiences $U(D_k^l)$ is firstly selected from the dataset D_k^l. This selection is performed by randomly choosing experiences $e_{k,j}$=< $\boldsymbol{s_{k,j}}$, $\boldsymbol{a_{k,j}}$, $r_{k,j}$, $\boldsymbol{s_{k,j}}$* > for j=1…J. Then, $Q_k(\boldsymbol{s_k},\boldsymbol{a_k},\theta_k)$ is updated based on the mini-batch gradient descent of $U(D_k^l)$. For this purpose, the average mean squared error (MSE) loss over all the J experiences $e_{k,j}$ in $U(D_k^l)$, denoted as $L(\theta_k)$, is firstly computed as:

$$L(\theta_k)= E_{e_{k,j} \in U(D_k^l)}[(r_{k,j}+\gamma\, max_{\boldsymbol{a_k}'}\, Q_k(\boldsymbol{s_{k,j}}^*,\boldsymbol{a_k}',\theta_k^-)- Q_k(\boldsymbol{s_{k,j}},\boldsymbol{a_{k,j}},\theta_k))^2] \quad (58)$$

Then, the mini-batch gradient descent of $L(\theta_k)$, denoted as $\nabla L(\theta_k)$, is obtained by differentiating $L(\theta_k)$ with respect to θ_k, which yields:

$$\nabla L(\theta_k)=E_{e_{k,j} \in U(D_k^l)}[\big(r_{k,j}+\gamma\, max_{\boldsymbol{a_k}'}\, Q_k(\boldsymbol{s_{k,j}}^*,\boldsymbol{a_k}',\theta_k^-)- Q_k(\boldsymbol{s_{k,j}},\boldsymbol{a_{k,j}},\theta_k)\big)\nabla_\theta Q_k(\boldsymbol{s_{k,j}},\boldsymbol{a_{k,j}},\theta_k)] \quad (59)$$

Finally, the weights in the $Q_k(\boldsymbol{s_k},\boldsymbol{a_k},\theta_k)$ network are updated according to:

$$\theta_k \rightarrow \theta_k + \tau \nabla L(\theta_k) \tag{60}$$

where τ is the learning rate. After each update of θ_k, the obtained $Q_k(\boldsymbol{s_k},\boldsymbol{a_k},\theta_k)$ can be used by the ε-Greedy policy to trigger actions in the network environment. Moreover, during the update of weights θ_k, the DQN agent has a counter m of the number of time steps since the last target DNN update and, when $m=M$, the weights in the target DNN are updated as $\theta_k^- = \theta_k$ and m is initialised again.

4.4.5. Capacity Sharing computation

In the considered MARL approach, the DQN agents associated to the different tenants in the system interact with a common network environment in a collaborative and coordinated manner in order to trigger the actions that the capacity sharing computation module will use to configure the capacity share $\boldsymbol{\sigma_k(t)}$ of each tenant.

Initially, the capacity shares of each tenant in each of the cells $\sigma_{k,n}(t=0)$ are initialised proportionally to the $SAGBR_k$ of the tenant, according to:

$$\sigma_{k,n}(t=0) = SAGBR_k / \left(\sum_{k'=1}^{K} SAGBR_{k'} \right) \tag{61}$$

Then, $\sigma_{k,n}(t)$ is dynamically tuned according to the selected actions $\boldsymbol{a_k(t)}$ by each agent at each time step. Even though the process of data collection by each DQN agent is performed independently of the others, the different DQN agents trigger their actions and store the experiences synchronously. At each time step t, each agent k is provided with its state $\boldsymbol{s_k(t)}$ and accordingly selects an action $\boldsymbol{a_k(t)}$ as previously explained. Next, the capacity sharing computation module gathers the selected actions $\boldsymbol{a_k(t)}$ by all the agents and computes the resulting capacity share solution $\boldsymbol{\sigma_k(t)}$. Specifically, the capacity share of tenant k at cell n, $\sigma_{k,n}(t)$, is updated according to:

$$\sigma_{k,n}(t) = \begin{cases} \sigma_{k,n}(t\text{-}1) + a_{k,n}(t) & \text{if } 0 \leq \sigma_{k,n}(t\text{-}1) + a_{k,n}(t) \leq \dfrac{MCBR_{k,n}}{c_n} \\ \sigma_{k,n}(t\text{-}1) & \text{Otherwise} \end{cases} \tag{62}$$

The formulation of (62) assures that $\sigma_{k,n}(t)$ is within its bounds and considers that the last capacity share value $\sigma_{k,n}(t\text{-}1)$ is maintained when $a_{k,n}(t)$ forces $\sigma_{k,n}(t)$ to be out of its bounds.

In some special situations, the fact that the actions of each tenant are triggered independently can lead to capacity sharing solutions that excess the total cell capacity in some cells (i.e. $\sum_{k=1}^{K} \sigma_{k,n}(t) > 1$). When this occurs in any cell, the capacity sharing computation module obtains the capacity shares in the cell by executing *Algorithm* 3. The algorithm firstly applies the actions of tenants aiming at decreasing or maintaining the capacity share in the cell (i.e., $a_{k,n}(t) \in \{-\Delta, 0\}$) and computes the resulting available cell capacity $\sigma_n^A(t)$ (lines 1-2). If there is no available capacity (i.e., $\sigma_n^A(t)=0$), the actions of tenants willing to increase its capacity in the cell are not applied (lines 3-4). In turn, when there is available capacity (i.e.,

$\sigma_n^A(t)>0$), the capacity share is obtained by distributing $\sigma_n^A(t)$ among those tenants with $a_{k,n}(t)= \Delta$ proportionally to their $SAGBR_k$ value as long as they are not already provided with more than $SAGBR_k$ (lines 5-6) according to:

$$\sigma_{k,n}(t)=\sigma_{k,n}(t\text{-}1)+\frac{\sigma_n^A(t)\cdot SAGBR_k}{\sum_{\substack{k'=1 \\ a_t(k',n)=\Delta, \\ T_t(k')\leq SAGBR_{k'}}}^{K} SAGBR_{k'}} \tag{63}$$

Algorithm 3. Capacity sharing computation for cell *n* for cell capacity excess situations

1 Compute $\sigma_{k,n}(t)$ of tenants with $a_{k,n}(t) \in\{-\Delta,0\}$ according to (62).
2 Compute available cell capacity share $\sigma_n^A(t)$ according to (47).
3 **If** $\sigma_n^A(t)=0$
4 Set $\sigma_{k,n}(t)= \sigma_{k,n}(t\text{-}1)$ of tenants with $a_{k,n}(t)=\Delta$.
5 **Else**
Compute $\sigma_{k,n}(t)$ by distributing $\sigma_n^A(t)$ among tenants with
6 $a_{k,n}(t)=\Delta$ and $T_k(t)\leqslant SAGBR_k$ proportionally to their $SAGBR_k$ according to (63).
7 **End if**

4.5. Performance metrics

In this section, different KPIs are defined in order to assess the performance of the model:

- Assigned capacity to tenant k at time step t ($A_k(t)$): It is measured in bps and is obtained from the capacity share $\boldsymbol{\sigma_k(t)}$ provided by the SON function and the capacity of each cell c_n as:

$$A_k(t) = \sum_{n=1}^{N} c_n \cdot \sigma_{k,n}(t) \tag{64}$$

- Average reward of tenant k (R_k): It is computed as the average of the reward $r_k(t)$ obtained by the tenant over a duration of G time steps.

$$R_k=\frac{1}{G}\sum_{t=0}^{G\text{-}1} r_k(t) \tag{65}$$

- SLA satisfaction ratio ($ss_k(t)$): It measures the ratio between the provided throughput $T_k(t)$ to tenant k and the minimum between the aggregated offered load of the tenant $O_k(t)$ and its $SAGBR_k$ value that is:

$$ss_k(t) = \min\left(\frac{T_k(t)}{\min(O_k(t),\, SAGBR_k)},1\right) \tag{66}$$

which ranges $0\leq ss_k(t)\leq 1$, taking $ss_k(t)=0$ value when the SLA is not satisfied and $ss_k(t)$ when it is fully satisfied. Note that the definition of $ss_k(t)$ considers that when $O_k(t)$ is lower than the $SAGBR_k$, $O_k(t)$ needs to be provided, whereas in the case that $O_k(t)$ is greater than $SAGBR_k$, at least $SAGBR_k$ needs to be provided.

- Average SLA satisfaction of tenant k (SS_k): It measures the average of SLA satisfaction ratio $ss_k(t)$ over a duration of G time steps, that is:

$$SS_k = \frac{1}{G}\sum_{t=0}^{G-1} ss_k(t) \tag{67}$$

- Average system utilization (U): It is computed as the average ratio between the aggregated throughput provided to all tenants and the total system capacity C over a duration of G time steps, that is:

$$U = \frac{1}{G}\sum_{t=0}^{G-1}\frac{1}{C}\sum_{k=1}^{K} T_k(t) \tag{68}$$

4.6. Performance evaluation

This section provides the performance evaluation of the proposed SON function for capacity sharing when considering diverse multi-tenant scenarios with different number of cells and diverse traffic distribution among cells. First, in Section 4.6.1 the behaviour of SON function is discussed under a single-cell scenario and, then, it is compared to two reference capacity sharing solutions, which highlights the benefits of the proposed solution. Second, the SON function is assessed in a multi-cell scenario with homogeneously distributed traffic among cells in Section 4.6.2. In this scenario, the capability of the agent of one tenant to learn a general policy applicable to any other tenant in the system is firstly assessed. This is followed by a discussion of the scalability of the solution by adding a new tenant in the considered scenario for evaluation. Afterwards, the impact of different model parameter configurations on the achieved performance and an analysis on the optimality of the SON function are provided. Finally, the robustness of the SON solution to deal with heterogeneities of the traffic distributions in multi-cell environments is analysed in Section 4.6.2 by considering diverse levels of spatial and temporal heterogeneity in the traffic distributions of the tenants in a multi-cell scenario and by analysing the impact of the SLA parameters in these situations.

4.6.1. Performance under a single-cell scenario

4.6.1.1. Scenario under consideration

The considered scenario is composed of a single cell that provides service to two different tenants, denoted *Tenant* 1 and *Tenant* 2. The scenario is characterised and configured as indicated in Table XVI.

The training has been performed by considering a training dataset for the DQN agents to learn how to behave under different traffic load conditions. This dataset is composed of 330 synthetic offered load patterns of the two tenants during a day in the cell of Table XVI, which has been proved to be sufficient for the training of the model for the considered scenario. In each of the patterns, the offered load is computed in periods of 3 minutes, so each pattern consists of 480 steps. The different patterns of each tenant have been combined in order that the DQN agents can visit several states so that the learning can be achieved. The simulation parameters for the training of the DQN agents are detailed in Table XVII. The results have been obtained by developing the whole model with the library *TF-Agents* [84].

TABLE XVI. SCENARIO CONFIGURATION.

Parameter		Cell configuration
PRB Bandwidth (B_n)		360 kHz
Number of available PRBs (W_n)		51 PRBs
Average spectral efficiency (S_n)		5 b/s/Hz
Total cell capacity (c_n)		91.8 Mb/s
$SAGBR_k$	Tenant 1	55Mb/s (corresponding to 60% of available capacity)
	Tenant 2	36Mb/s (corresponding to 40% of available capacity)
$MCBR_{k,n}$	Tenant 1	73 Mb/s (corresponding to 80% of available capacity)
	Tenant 2	

TABLE XVII. SIMULATION PARAMETERS FOR TRAINING.

Parameter	Value
Initial collect steps	2000
Maximum number of training steps for learning	160000
Experience Replay buffer maximum length (l)	100000
Mini-batch size (J)	100
Discount factor(γ)	0.9
Learning rate (τ)	0.001
ε value (ε-Greedy)	0.1
DNN configuration	Single layer with 100
Time step duration (Δt)	3 min
Action step (Δ)	0.03

4.6.1.2. SON function behaviour

The evaluation of the trained model has been performed by considering two situations, denoted as *Situation A* and *Situation B*, with the different offered load patterns included in Fig. 32 and Fig. 33, respectively, which show the temporal evolution of the aggregated offered load $O_k(t)$ by the two tenants in the cell during a day. *Situation A* corresponds to an offered load pattern where *Tenant* 1 requires more capacity than *Tenant* 2 during the morning while the contrary case is given during the afternoon. *Situation B* presents the contrary case, where the roles of *Tenant* 1 and *Tenant* 2 are reversed. Notice that, in *Situation A*, the aggregated load by both tenants does not exceed the total cell capacity (c_n=91.8Mb/s) at any time while in *Situation B* the total capacity is exceeded for a long period of time.

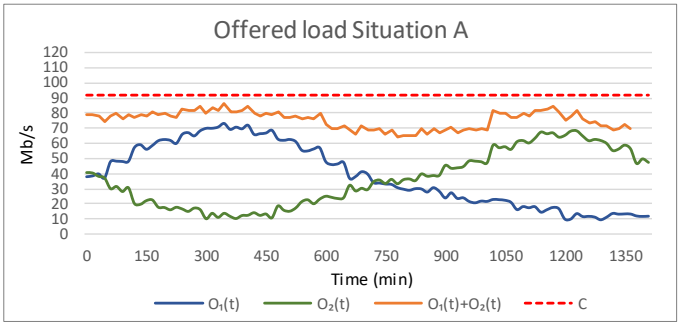


Fig. 32. Offered load of *Tenant* 1 and *Tenant* 2 in *Situation A*.

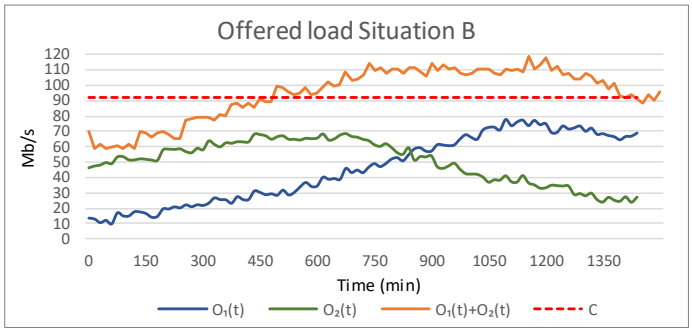


Fig. 33. Offered load of *Tenant* 1 and *Tenant* 2 in *Situation B*.

Fig. 34 and Fig. 35 compare the assigned capacity $A_k(t)$ with the offered load $O_k(t)$ and the $SAGBR_k$ for *Tenant* 1 and *Tenant* 2 in *Situation A* and *Situation B*, respectively. In *Situation A*, the offered load of both tenants is generally served as there is enough capacity to fulfil the both of them. The capacity sharing mechanism provides the demanded capacity to both tenants, including those cases when the offered loads of *Tenant* 1 and *Tenant* 2 exceed $SAGBR_1$ and $SAGBR_2$, respectively, making efficient use of the available capacity and exhibiting the capability to exploit the complementarities in the traffic profiles between both tenants. In *Situation B*, similar behaviour is observed but lower assigned capacity than offered is obtained during the periods where more capacity than available is requested. In those periods, the required capacity is given to the tenants whose offered load $O_k(t)$ is lower than $SAGBR_k$, which shows that the capacity sharing function assures the $SAGBR_k$ established in the SLA.

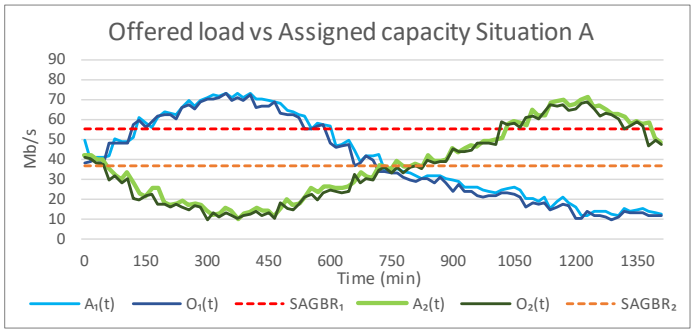


Fig. 34. Offered load $O_k(t)$ vs assigned capacity $A_k(t)$ in *Situation A*.

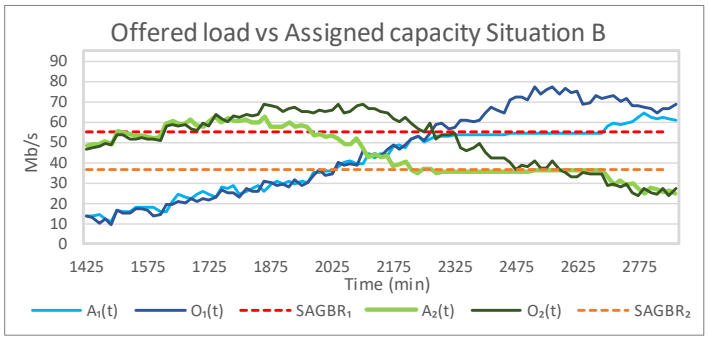


Fig. 35. Offered load $O_k(t)$ vs assigned capacity $A_k(t)$ in *Situation B*.

4.6.1.3. Comparison of SON function with reference capacity sharing solutions

For benchmarking purposes, the performance obtained in *Situation B* by the DRL-based capacity sharing solution has been compared to two reference capacity sharing solutions: *Reference* 1, which considers $SAGBR_k$ as the maximum capacity that can be assigned to tenant k in any case, and *Reference* 2, which

considers that both tenants share the overall capacity independently on $SAGBR_k$ and in the case that more capacity than the available is requested, the capacity is distributed among tenants according to their offered load. Fig. 36 and Fig. 37 show the cumulative density function (CDF) of SLA satisfaction ratio ss_k for *Tenant* 1 and *Tenant* 2, respectively. While *Reference* 1 always fulfils the SLA, *Reference* 2 presents much lower SLA satisfaction, given that both tenants are provided with lower throughput than required when the offered load exceeds the total capacity. The proposed MARL approach is able to improve the SLA satisfaction of Reference 2 by serving all the offered load $O_k(t)$ as long as it is lower than $SAGBR_k$.

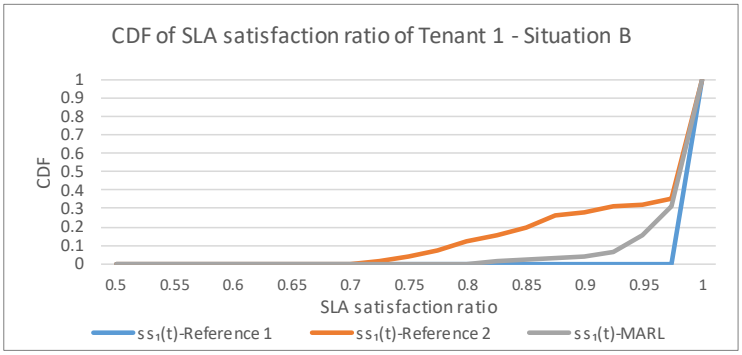


Fig. 36. CDF of SLA satisfaction ratio of *Tenant* 1 in *Situation B*.

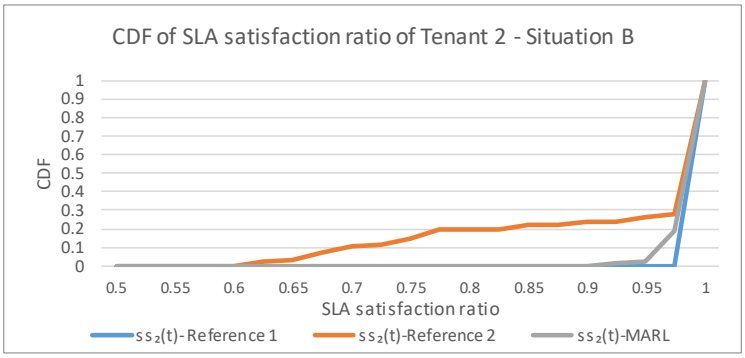


Fig. 37. CDF of SLA satisfaction ratio of *Tenant* 2 in *Situation B*.

Moreover, Fig. 38 compares the CDF of the percentage of the system utilisation of the three approaches. The system utilisation is defined as the ratio between the aggregate throughput of the two tenants and the total cell capacity. *Reference* 1 presents the lowest system utilisation, as no more than $SAGBR_k$ is provided even though there is enough capacity unused in the system to satisfy the offered load $O_k(t)$. However, the proposed MARL approach substantially improves the utilisation, with a performance very close to *Reference* 2. These results show how the presented approach allows satisfying the SLA and making efficient use of the resources, achieving a good trade-off between the two benchmarking schemes.

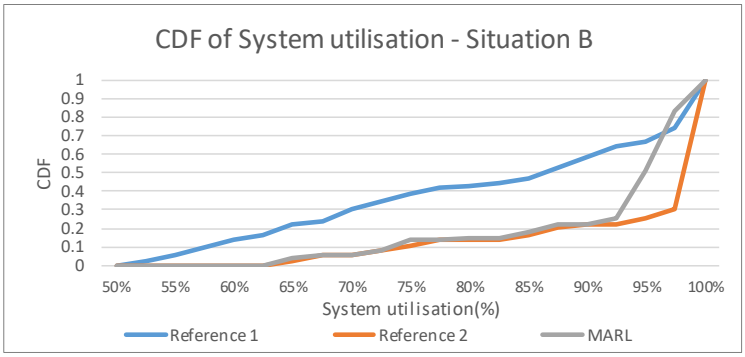


Fig. 38. CDF of overall system utilization of in *Situation B.*

4.6.2. Performance under a multi-cell scenario with homogeneous traffic distribution among cells

4.6.2.1. Scenario under consideration

The assumed scenario comprises a NG-RAN infrastructure with five cells that serve the users of two different tenants, denoted as Tenant 1 and Tenant 2. The configuration of the scenario is presented in Table XVIII, including the cells configuration and the SLA parameters established for each tenant.

TABLE XVIII. SCENARIO CONFIGURATION.

Parameter		Value
Number of tenants (K)		2
Number of cells (N)		5
PRB Bandwidth (B_n)		360 kHz
Number of cell available PRBs (W_n)		65 PRBs
Average spectral efficiency (S_n)		5 b/s/Hz
Total cell capacity (c_n)		117 Mb/s
Total system capacity (C)		585 Mb/s
$SAGBR_k$	Tenant 1	351Mb/s (corresponding to 60% of C)
	Tenant 2	234Mb/s (corresponding to 40% of C)
$MCBR_{k,n}$	Tenant 1	93.6 Mb/s (corresponding to 80% of c_n)
	Tenant 2	

The model has been trained according to the parameters of Table XIX. The dataset considered for training is composed of 1400 synthetically generated offered load patterns of Tenant 1 and Tenant 2 in the different cells during one day, considering different combinations of $SAGBR_k$ values for both tenants.

After the model has been trained, the resulting policies π_k are evaluated using the offered load patterns shown in Fig. 39. The figure plots the aggregated offered loads among all the cells of Tenant 1, $O_1(t)$, and Tenant 2, $O_2(t)$, during one day. The figure also includes the values of $SAGBR_1$ and $SAGBR_2$, the total system capacity C and the aggregated offered loads of both tenants $O(t)$. Note that the offered loads of both tenants exceed their $SAGBR_k$ at some point during the day and the system offered load $O(t)$ is higher than C during the time period from 900 min to 1300 min. Moreover, a uniform distribution of the load among the different cells has been considered.

TABLE XIX. DRL-BASED MODEL PARAMETERS.

Parameter		Value
Initial collect steps		5000
Maximum number of time steps for training		$2 \cdot 10^6$
Experience Replay buffer maximum length (l)		10^7
Mini-batch size (J)		256
Learning rate (τ)		0.0001
Discount factor(γ)		0.9
ε value (ε-Greedy)		0.1
DNN configuration		Input layer: 17 nodes 1 full connected layer: 100 nodes Output layer: 243 nodes
Reward weights	φ_1	0.5
	φ_2	0.4
Time step duration	Δt	3 min
Action step	Δ	0.03

Fig. 39. Offered loads of Tenant 1 and 2 during a day.

Regarding the complexity of the proposed approach, it has been assessed in terms of the execution time of the MARL approach over a machine with 2 CPU AMD Opteron 4386 operating with Ubuntu 18.04, configured to use 2 cores and 8G RAM. Specifically, the execution of one trained DQN agent in one time step during evaluation stage lasts 3.8 ms on average, which is a sufficiently low value that would enable the operation in a practical system. Concerning the training stage, it is obtained that the execution of a time step lasts approximately 36 ms. The larger duration of the time step in the training stage compared to the evaluation stage is motivated by the additional operations required during the training for updating the weights of the DNNs.

4.6.2.2. Generalization of the learnt policies

According to the proposed approach, the DQN agent of each tenant learns its own policy during the training and then this policy is applied during the evaluation. However, considering that the training of the different tenants has been done under very different situations of their own load and the load of the others and for different SLA parameters, in this section, we intend to analyse to what extent there are significant differences between the policies learnt by the different tenants. In this way, the main goal is to assess

whether it is possible or not to generalise a policy leant by one tenant so that it can be also used by another tenant.

To conduct the analysis, the capacity sharing solution for the offered loads of Fig. 39 is obtained under two different policy application modes. In *Mode A*, the DQN agent of each tenant applies its trained policy, i.e., the DQN agent of Tenant 1 applies policy π_1, and the DQN agent of Tenant 2 applies policy π_2. Both policies are the ones obtained after $200 \cdot 10^4$ training steps. In turn, *Mode B* considers that the DQN agents of both Tenant 1 and Tenant 2 apply the same policy π_1 learnt for Tenant 1.

Fig. 40 presents the temporal evolution of the offered load of Tenant 2 against its assigned capacity $A_2(t)$ for policy application *Mode A* and *Mode B*. In addition, the $SAGBR_2$ of Tenant 2 is included. The assigned capacity for both policy application modes generally adapts to the offered load for all the situations where the total offered load $O(t)$ (seen in Fig. 39) does not exceed the system capacity C. In turn, when $O(t)$ exceeds the system capacity, the assigned capacity to Tenant 2 is kept in the $SAGBR_2$ value. The figure shows that very little differences are observed in the assigned capacity $A_2(t)$ when applying the policies according to *Mode A* and *Mode B*.

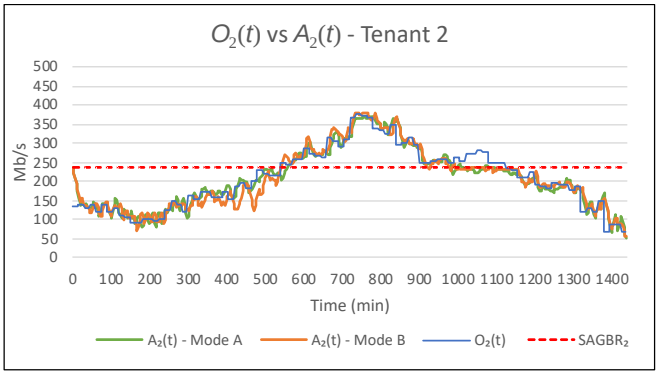


Fig. 40. Offered load vs assigned capacity for Tenant *2* for tenant policy application *Mode A* and *B*.

Moreover, to quantitatively assess the differences between both modes, Table XX provides the average reward R_k and the SLA satisfaction SS_k for Tenant 1 and Tenant 2 in addition to the average system utilisation U. The obtained values show that the achieved performance for both policy application modes is very similar, with differences lower than 1% for all the analysed KPIs.

TABLE XX. KPIS FOR BOTH POLICY APPLICATION MODES.

Policy application mode		**Mode A**	**Mode B**
Average reward	**Tenant 1 (R_1)**	0.9673	0.9674
	Tenant 2 (R_2)	0.9541	0.9483
SLA Satisfaction	**Tenant 1 (SS_1)**	0.9725	0.9742
	Tenant 2 (SS_2)	0.9705	0.9577
Average utilisation (U)		0.8885	0.8861

It is worth pointing out that for the case of Tenant 1 there are some slight differences in Table XX although the policy applied for this tenant is the same in both modes. The reason is that the actions taken by the DQN

agent of Tenant 2, which can be slightly different when changing the applied policy, impact on the state seen by the DQN agent of Tenant 1 and, thus, on the selected actions by this tenant. In any case, the impact of these different actions on the performance is negligible as seen in the table.

Concerning Tenant 2, the reason that similar behaviour is obtained for policy application *Mode A* and *Mode B* is that, although the training of policies π_1 and π_2 has been performed independently for both tenants, this has considered a dataset composed of several offered load situations, exploring different complementarities between the two tenants, jointly with diverse combinations of $SAGBR_k$ and $MCBR_{k,n}$ values. This allows the agents to learn equivalent policies that can be generalised to many offered load situations and SLA requirements. This observation has important positive implications on the practicality of the proposed approach, because it means that a single training process carried out by one DQN agent using a dataset that covers a wide range of offered load situations and SLA requirements can be sufficient to obtain a policy that is valid for multiple tenants. As a consequence, a reduction of the complexity of the training process will be achieved in a multi-agent scenario. Moreover, this also facilitates the scalability of the model to add new tenants in the scenario, because the addition of a tenant can be done without retraining the previous learnt policies, as it will be studied in the next sub-section.

4.6.2.3. Addition of a new tenant

Following the generalization capability of the trained policies that has been observed in previous section, this section aims at assessing the association of already trained policies to new tenants that are added in the scenario, without neither training new policies for the new tenants nor retraining (i.e., training again) the policies from the existing tenants. To this end, a new tenant, denoted as Tenant 3, is introduced to the scenario of Table XVIII. Instead of performing a separate training for the new Tenant 3, the previously trained policy for Tenant 1, π_1, is used for this new tenant as well as for Tenant 1 and 2. Since the $SAGBR_k$ of Tenants 1 and 2 use the total system capacity of Table XVIII, in order to support the new tenant, the capacity in the system is extended by increasing the cell bandwidth from 25 MHz to 30 MHz. As a result, the number of PRBs in each cell is increased to W_n=78 PRBs, providing a total cell capacity c_n =143.2 Mb/s and, thus, a total system capacity of C =716 Mb/s. The SLA established for Tenant 3 considers $SAGBR_3$=93.6 Mb/s and $MCBR_{3,n}$=114.56 Mb/s, corresponding to 80% of the cell capacity. The $SAGBR_k$ of Tenant 1 and 2 remain the same as in Table XVIII, whereas the $MCBR_{k,n}$ of those tenants is updated to $MCBR_{1,n}$=$MCBR_{2,n}$= 114.56Mb/s given that the cell capacity has increased.

The offered loads considered for evaluation during one day are plotted in Fig. 41 together with the $SAGBR_k$ values, the aggregated offered load in the system $O(t)$ and the total system capacity C. The offered loads of Tenant 1 and Tenant 2, $O_1(t)$ and $O_2(t)$, are the same as in Fig. 39, and the offered load of Tenant 3, $O_3(t)$, presents lower values than the other tenants, reaching its higher values at t=570 min and t=880 min when its $SAGBR_3$ is exceeded. Despite introducing Tenant 3, the total offered load of the three tenants only slightly exceeds the system capacity from t=1000 min to t=1200 min.

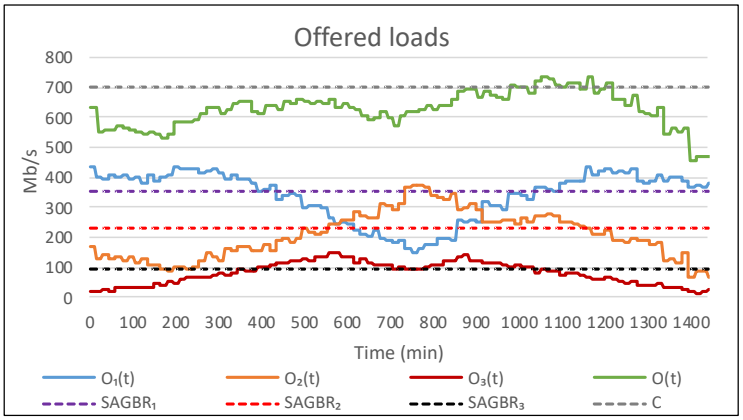


Fig. 41. Offered load of Tenant 1, 2 and 3 during a day.

Fig. 42 shows the offered loads $O_k(t)$ against the assigned capacity $A_k(t)$ of Tenant 1, 2 and 3, in addition to their $SAGBR_k$ values. Since most of time there is enough capacity to fulfil the offered load of the three tenants, the offered loads are satisfied nearly all the day. When the overall offered load $O(t)$ exceeds the system capacity, the tenants that required more capacity than their $SAGBR_k$ are assigned with lower capacity than their offered load, such as Tenant 2 from t=1035 min to t=1115 min. In the case of Tenant 3, the offered load $O_3(t)$ is generally satisfied since in the periods when the $O_3(t)$ is larger than $SAGBR_3$, there is enough capacity in the system to satisfy all the tenants. These results show qualitatively that the policy learnt by one tenant is general enough to properly assign the capacity to the other tenants according to their offered loads and SLA requirements and, additionally, performs satisfactorily in front of changes in the system capacity, since the internal parameters of the DQN agent (i.e., state, reward factors, actions, etc.) are defined in relative values. This highlights the capability of scaling of the proposed solution, as the trained policies can be used by new tenants in the scenario without the need of performing a separated training for each of tenants or the need of re-training when the capacity in the system changes. These results also provide evidence of the advantage of a multi-agent approach with respect to a single-agent approach, where a single agent manages all the tenants and the addition of a new tenant would imply the training of the whole solution again. Moreover, this would require a larger training duration due to the larger state and action spaces resulting from the additional state and action dimensions for the new tenant. Instead, in the proposed approach, a new tenant can be added just by associating an already trained policy.

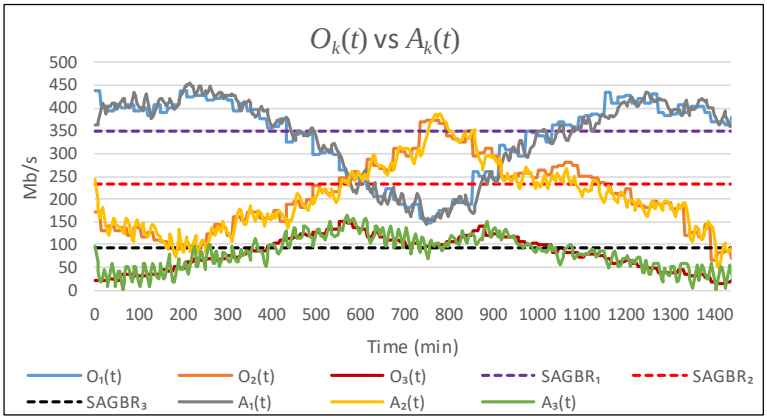


Fig. 42. Offered load vs assigned capacity for Tenant 1, 2 and 3.

To further illustrate the scaling capability of the proposed approach, the performance achieved when applying the policy π_1 for all tenants is compared against the case of applying the policies π_1, π_2 and π_3 specifically trained for each tenant. To this end, policies π_1, π_2 and π_3 have been trained according to model parameters detailed in Table XIX by considering the scenario with Tenant 3 and extended capacity. Table XXI includes the resulting average SLA satisfaction SS_k and the average reward for Tenant 1, 2 and 3 and the average system utilisation U when applying the specifically trained polices for each tenant and when applying the policy π_1. Once again, the achieved values for the different assessed indicators for both cases are really close, presenting differences below 1.5% for all indicators. This highlights the scaling capability of the proposed approach since the application of a previously trained policy is able to adapt to the offered loads of the new tenant and the loads of the other tenants when the system capacity increases.

Regarding the achieved values for the different analysed performance indicators in Table XXI, although high values are obtained for all the tenants, Tenant 1 achieves the highest, closely followed by Tenant 2, and Tenant 3 presents the lowest values. The reason is that the offered load values of Tenant 3 are much lower than the ones for Tenant 1 and Tenant 2, and thus, the analysed performance indicators are more affected by the increases and decreases in steps of Δ. This means that decreasing the assigned capacity of Tenant 1 by Δ=0.03 has a lower impact on R_k and SS_k of Tenant 1 than for Tenant 3. Since the offered load levels of Tenant 2 are similar to the ones of Tenant 1, lower differences are obtained between them as a result of this effect. Therefore, in order to achieve higher performance values for Tenant 3, lower values of Δ would be more appropriate since they would have a lower impact on the assessed parameters. This reveals that the selection of Δ needs to jointly consider the traffic type and levels of all tenants in order to choose a value that best satisfies all of them given the clear impact of Δ on the achieved performance, as is studied in the following section.

TABLE XXI. PERFORMANCE PARAMETERS.

Performance metrics		Tenant- specific policies	Tenant 1 policy
Average reward	**Tenant 1 (R_1)**	0.964	0.967
	Tenant 2 (R_2)	0.939	0.949
	Tenant 3 (R_3)	0.873	0.859
SLA Satisfaction	**Tenant 1 (SS_1)**	0.986	0.979
	Tenant 2 (SS_2)	0.957	0.961
	Tenant 3 (SS_3)	0.901	0.893
Average utilisation (U)		0.843	0.845

4.6.2.4. Impact of action and time step

This section aims at analyzing the impact of the values of time step duration Δt and action increment/decrement step Δ on the proposed SON function. For this purpose, the policy π_1 of Tenant 1 has been trained in the scenario of Table XVIII by considering different parameter configurations. The assessed configurations include all the combinations between time step duration Δt={1, 3, 5, 15} min and action

step Δ={0.01, 0.02, 0.03, 0.05, 0.07, 0.09}. The rest of the parameters used for training are those specified in Table XIX .

First, the impact Δt and Δ on the training evolution process is studied. To this end, the policy π_1 for Tenant 1 obtained by the training every 10^4 time steps has been evaluated by applying it on the offered load of Tenant 1 of Fig. 39. This allows capturing how the training process progressively updates the learnt policy when increasing the number of training steps. Fig. 43 and Fig. 44 show the evolution of the average reward R_1 of Tenant 1 for Δt={1, 3, 5, 15} min when considering, respectively, a small action step Δ=0.01 and a large action step Δ=0.07. For both values of Δ, higher average reward is achieved when reducing Δt, since the policy is triggered more frequently and thus it can more easily react to changes. However, evident differences are observed between the training evolution when using Δ=0.01 and Δ=0.07. For Δ=0.07, the average reward presents an initial period of around $50 \cdot 10^4$ training time steps where it increases and presents high fluctuations until the average reward stabilises to a value in the range between 0.94 and 0.96 depending on Δt and the fluctuations decrease drastically, reflecting that the algorithm has achieved convergence. Instead, for Δ=0.01, the average reward keeps increasing during the whole analysed training period and it exhibits larger fluctuations than for Δ=0.07, so convergence takes longer. The reason is that with low values of Δ, the actions at each time step have a low impact on the next state and the reward obtained, so the training has more difficulties to converge.

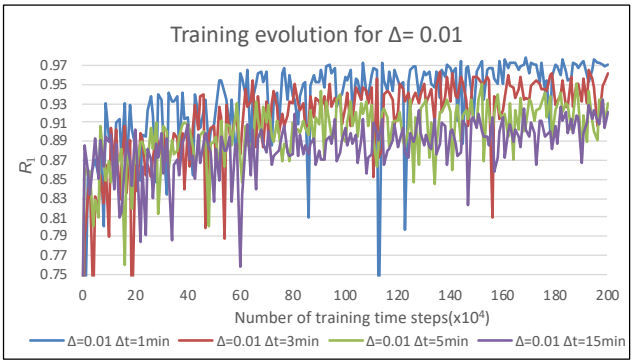


Fig. 43. Average aggregated reward every 10000 time steps during the training for Δ=0.01 and Δt={1,3,5,15} min.

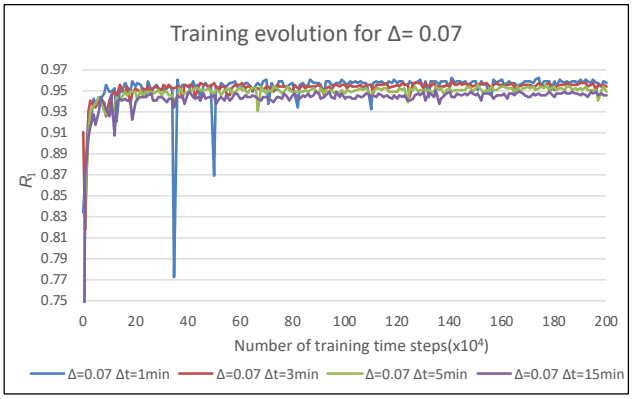


Fig. 44. Average aggregated reward every 10000 time steps during the training for Δ=0.07 and Δt={1,3,5,15} min.

Fig. 45 depicts the average reward R_1 of Tenant 1 obtained between $100 \cdot 10^4$ and $200 \cdot 10^4$ training time steps for the different analysed combinations of Δt and Δ. The highest reward is observed for Δt=1 min and the

reward decreases when increasing Δ*t*. Regarding the effect of the action step Δ, the highest reward is achieved for Δ=0.02 for Δ*t*=1 min while the maximum is achieved for Δ=0.03 for the rest of time step durations and, for higher values than Δ=0.03, a decreasing trend is observed. This occurs since large values of Δ make more difficult the adjustment of the assigned capacity to the offered load of Tenant 1 in Fig. 39. These results together with the fluctuations obtained in Fig. 43 and Fig. 44 reveal that a trade-off exists when selecting the value of Δ: a higher reward is achieved for low values of Δ but at the cost of slower convergence. In addition, Fig. 45 shows that the combination of large values of Δ*t* with low values of Δ (e.g., Δ=0.01 and Δ*t*=15 min) lead to poor performance as those combinations do not allow adapting to the dynamics of the offered load.

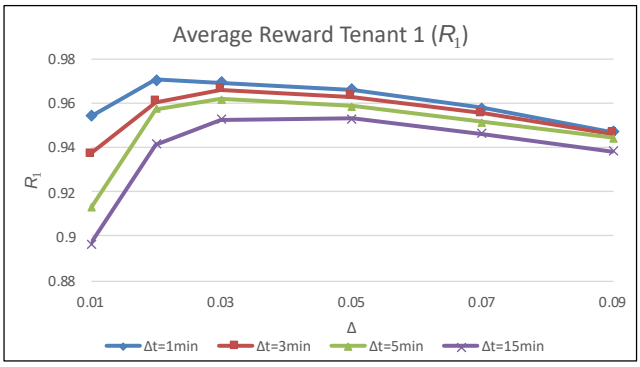


Fig. 45. Average aggregated reward between $100 \cdot 10^4$ and $200 \cdot 10^4$ training time steps for the different configurations.

Finally, the impact of Δ and Δ*t* on the performance metrics is analysed in Table XXII. It presents the average SLA satisfaction SS_k and system utility *U* obtained for different values of Δ and Δ*t* when applying the trained policies for Tenant 1 after $200 \cdot 10^4$ training time steps to the offered loads of Tenant 1 and Tenant 2 in Fig. 39. Focusing on the average SLA satisfaction, the capacity sharing SON function achieves high values above 0.9 for both tenants and all the combinations of Δ and Δ*t*. Higher values are generally obtained for combinations with low values of Δ and Δ*t*, since they adapt better to the changes in the offered loads. Also, a decreasing trend in the SLA satisfaction is observed when increasing Δ for values of Δ beyond 0.03. Comparing the SLA satisfaction obtained for Tenant 1 and Tenant 2, slight differences are obtained between them. About the impact of Δ and Δ*t* on the average system utilization *U*, similar effects than in the case of the SLA satisfaction are obtained, achieving the highest values for Δ=0.02 for all values of Δ*t* in exception of Δ*t*=15 min, which the maximum is achieved for Δ=0.03, and being the utilization reduced when increasing Δ beyond the maximum.

Based on the obtained results, it is concluded that the selection of the Δ and Δ*t* values has a clear impact on the training evolution of the policies and their achieved performance and an adequate selection of these values that jointly considers the specific traffic dynamics of the different tenants is fundamental for ensuring an accurate learning process and a good compromise between the different KPIs achieved by the capacity sharing SON function.

TABLE XXII. KPIS FOR THE DIFFERENT COMBINATIONS OF Δ *AND* Δ*T*.

Action step value (Δ)	**Average SLA satisfaction**								**Average system Utilisation (*U*)**			
	Tenant 1 (SS_1)				**Tenant 2 (SS_2)**							
	Δ*t*=1min	**Δ*t*=3min**	**Δ*t*=5min**	**Δ*t*=15min**	**Δ*t*=1min**	**Δ*t*=3min**	**Δ*t*=5min**	**Δ*t*=15min**	**Δ*t*=1min**	**Δ*t*=3min**	**Δ*t*=5min**	**Δ*t*=15min**
0.01	0.972	0.972	0.960	0.933	0.961	0.976	0.958	0.971	0.877	0.879	0.848	0.823
0.02	0.984	0.978	0.982	0.956	0.972	0.976	0.974	0.969	0.897	0.894	0.884	0.855
0.03	0.982	0.974	0.973	0.975	0.973	0.958	0.966	0.979	0.894	0.886	0.881	0.883
0.05	0.981	0.973	0.973	0.960	0.956	0.944	0.946	0.958	0.885	0.878	0.873	0.856
0.07	0.963	0.970	0.968	0.959	0.934	0.935	0.916	0.949	0.861	0.869	0.863	0.865
0.09	0.942	0.960	0.952	0.961	0.919	0.908	0.908	0.922	0.822	0.848	0.843	0.850

4.6.2.5. Optimality analysis

In this section, the optimality of the capacity sharing SON function is analyzed by comparing its performance to the optimum. The optimum has been obtained by an exhaustive search algorithm that evaluates in each time step all the possible values of capacity share $\boldsymbol{\sigma_k(t)}$ of Tenant 1 and Tenant 2, discretized in steps of Δ, and selects the one that achieves the maximum aggregate reward of both tenants. To assess the optimality in a wide range of offered load situations, results have been obtained for a set of 240 offered load temporal patterns of one-day duration, which include diverse offered load behaviours with diverse complementarities between the offered loads of Tenant 1 and Tenant 2. For each pattern, results have been obtained using the scenario configuration in Table XVIII and by applying the trained policy π_1 of Tenant 1 to both tenants. For each pattern, results are obtained in terms of the optimality ratio, which is defined as the average of the aggregate reward of Tenant 1 and Tenant 2 obtained with the SON function divided by the average optimum reward over all the time steps in an offered load pattern.

Fig. 46 presents the evolution of the optimality ratio during the training process for the offered load pattern of Fig. 39. This has been obtained by evaluating the policy π_1 every $5 \cdot 10^4$ training steps and computing the optimality ratio. It is observed that, initially, the optimality ratio increases abruptly with the number of training steps and, after approximately $5 \cdot 10^4$ training steps, it achieves values higher than 0.94. Then, it increases slowly with the number of training steps and stabilises to a value of around 0.97, corresponding to the situation when the algorithm has converged. The figure also reflects that no significant improvements are obtained by increasing the number of training steps beyond $50 \cdot 10^4$.

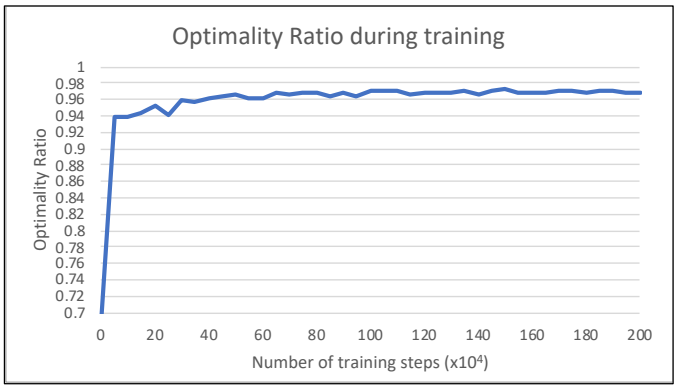


Fig. 46. Optimality ratio during training.

To analyze the optimality ratio under a broader range of situations, Fig. 47 shows the CDF of the optimality ratios obtained for the different offered load patterns with the policy learnt after $200 \cdot 10^4$ time steps. Results reveal that the optimality ratio for all the analysed offered load patterns range between 0.94 and 0.98. Moreover, it has been obtained that the average optimality ratio is 0.96. Overall, results reveal that the proposed MARL approach achieves a behaviour very close to the optimum and highlight the capability of the trained policy π_1 to adapt to diverse offered loads. It is also worth noting that this near optimal results are obtained with very small execution times of the trained policy, as previously discussed in Section 4.6.2.1, while the exhaustive search method requires to assess all the combinations for each time step, which is highly time consuming and requires execution times higher in several orders of magnitude than the MARL approach.

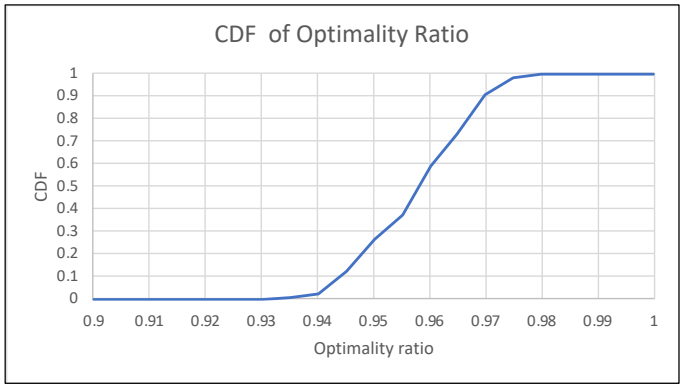


Fig. 47. Cumulative Density Function (CDF) of optimality ratio.

4.6.3. Performance under a multi-cell scenario with heterogeneous traffic distribution among cells

4.6.3.1. Scenario under consideration

The scenario considered for evaluation is comprised of a NG-RAN infrastructure in a 3 km x 3 km area with N=5 cells, which serve the users of K=2 tenants, denoted as *Tenant* 1 and *Tenant* 2. The scenario has been configured according to the parameters in Table XXIII, which includes the cells configuration, their position in the area under consideration and the SLA parameters of each tenant.

TABLE XXIII. SCENARIO CONFIGURATION.

Parameter		Value
Number of tenants (K)		2
Number of cells (N)		5
Area		3 km x 3 km
Cell position (km)	Cell n=1	(1.5,1.5)
	Cell n=2	(0.5, 2.5)
	Cell n=3	(2.5, 2.5)
	Cell n=4	(0.5,0.5)
	Cell n=5	(2.5,0.5)
PRB Bandwidth (B_n)		360 kHz
Number of PRBs per cell (W_n)		78 PRBs
Average spectral efficiency		5 b/s/Hz
Cell capacity (c_n)		140 Mb/s
Total system capacity (C)		700 Mb/s
$SAGBR_k$	Tenant 1	420 Mb/s (60% of system capacity)
	Tenant 2	280 Mb/s (40% of system capacity)
$MCBR_{k,n}$	Tenant 1	112 Mb/s (80% of cell capacity c_n)

To generate heterogeneous spatial and temporal distributions of the offered load of the two tenants in the different cells, it is assumed that at time step t the offered load density (Mb/s/km^2) of tenant k is spatially distributed according to the sum of a constant offered load density μ_k and a bivariate Gaussian distribution centred at the position $(x_k(t),y_k(t))$ with standard deviation d_k and offered load density in the center m_k. The center of the Gaussian distribution $(x_k(t),y_k(t))$ moves horizontally along the scenario with speed v_k. Then, the offered load of tenant k in cell n at time step t, $o_{k,n}(t)$, is obtained by aggregating the offered load density over the cell service area determined by the Voronoi tessellation.

Based on this methodology, the MARL model has been evaluated under four different offered load situations that reflect different levels of heterogeneity, denoted as *Situations* 1-4, whose configuration parameters have been included in Table XXIV. For each situation, the offered load of each tenant in each cell has been obtained during a day. The level of heterogeneity in the different situations is varied through the values of offered load density in the center m_k and deviation d_k. As a result, Situation 1 corresponds to a situation where the spatial distribution of the offered load of one tenant among the different cells is nearly homogeneous. Then, the level of heterogeneity is increased in Situations 2-4, being Situation 4 the one with the most unbalanced load among cells. To illustrate this, Fig. 48 plots the maps with the offered load densities for *Tenant* 1 and *Tenant* 2 in Situation 4 at some illustrative times. The black triangles indicate the positions of the 5 cells.

TABLE XXIV. CONFIGURATION OF OFFERED LOAD SITUATIONS.

Parameter		**Tenant 1**	**Tenant 2**
Initial position$(x_k(0),y_k(0))$ (km)		(0, 0.5)	(1.5, 2.5)
Speed (v_k) (km/h)		0.125	-0.29
Offered load density configuration (m_k(Mb/s/km^2), d_k(km))	Situation 1	(24, 5)	(16, 5)
	Situation 2	(28, 3)	(24, 3)
	Situation 3	(36, 1)	(36, 1)
	Situation 4	(72, 1)	(96, 0.5)
Constant offered load density (μ_k) (Mb/s/km^2)		20	16

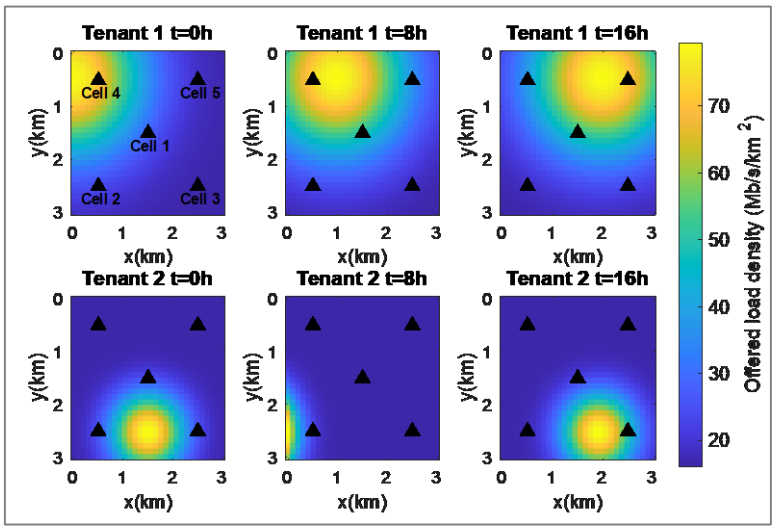


Fig. 48. Offered load density maps of Tenant 1 and 2 during a day.

The MARL model has been trained considering a time step duration Δt = 5 min and Δ = 0.03. The rest of DQN parameters have been configured according to Table XIX. The data used for the training has been generated by obtaining a wide range of spatial and temporal distributions of the offered loads of both tenants, obtained by modifying the values of the parameters of the Gaussian distributions of Tenant 1 and Tenant 2, including their initial position and speed, and diverse combinations of their SLA parameters.

4.6.3.2. Performance under different heterogeneity levels

In order to evaluate the capability of the policies learnt by the DQN agents of the two tenants to adapt to different levels of heterogeneity, they have been applied to the offered loads $o_{k,n}(t)$ in the Situations 1-4. Fig. 49 compares the resulting average offered load and the average assigned capacity (both expressed as a percentage of the cell capacity c_n) in the different cells for Tenant 1 and Tenant 2 in Situations 1-4. The aggregated offered load and the aggregated assigned capacity of each tenant at system level (i.e., among all cells) is also included as a percentage of the total capacity. Results reveal that the assigned capacity takes close values to the offered load requirements both at cell and system levels for the different situations, regardless of the level of heterogeneity. In fact, the obtained differences between the offered loads and the assigned capacities are lower than 8% for all cases, which are mainly due to the incremental action design, which makes that the assigned capacity fluctuates around the offered load within a margin between Δ and -Δ. The highest differences are observed for cell 4 in Situations 2 and 3 and for cells 2 and 5 in Situation 4, since their total offered load in these cells exceeds c_n during some periods, so the offered load of both tenants in those cells cannot be satisfied all the time (this can be seen when looking at the period between 19 hours and 22 hours of the second graph of Fig. 50 in the next section that depicts the cell offered load normalized to the cell capacity against the assigned capacity for cell 2). Moreover, results show that in certain cases when the traffic among cells is unbalanced and, in some cells, the offered load is higher than the relative $SAGBR_k$, the policy is able to support this load by smartly distributing the assigned capacity in accordance with the spatial traffic distribution. For example, the average offered load of Tenant 1 in Situation 4 in cells 4 and 5 exceeds the relative $SAGBR_k$ of 60% but the policy is able to support it since the offered load in the rest of cells is much lower than 60%. These results highlight the capability of the proposed solution to satisfactorily adapt to diverse levels of offered load heterogeneity among cells.

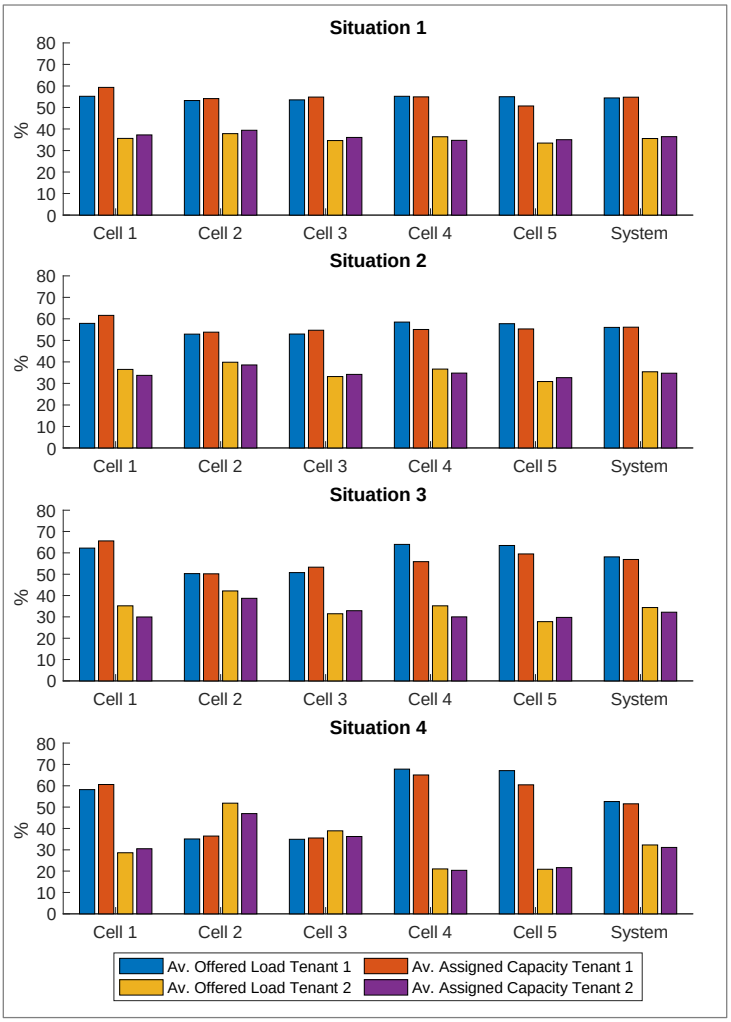


Fig. 49 Average offered load and assigned capacity per cell and at system level for each considered situation.

Table XXV includes the average reward, the average SLA satisfaction SS_k and the average assigned capacity ratio, which is defined as the ratio between the assigned cell capacity and the cell offered load over all the cells in the scenario. These metrics have been obtained for both tenants considering the average over one day. Results show that the learnt policies achieve high average reward for both tenants in all situations. The good performance is also reflected in the values of SLA satisfaction, which are higher than 0.96 for Tenant 1 and 0.93 for Tenant 2. The fact that slightly lower SLA satisfaction is obtained for Tenant 2 is that the traffic level of Tenant 2 is generally lower than that of Tenant 1, so the SLA satisfaction is more affected by the increases and decreases in action steps Δ than in the case of Tenant 1. Finally, in relation to the assigned capacity ratio, the obtained values are close to 1, with maximum deviations of 8%. This indicates that the assigned capacity properly matches the offered load with little overprovisioning.

TABLE XXV. SYSTEM-LEVEL PERFORMANCE INDICATORS.

Performance Indicator		Situation 1	Situation 2	Situation 3	Situation 4
Av. Reward	Tenant 1	0.96	0.97	0.96	0.96
	Tenant 2	0.95	0.94	0.94	0.94
Av. SLA satisfaction (SS_k)	Tenant 1	0.97	0.98	0.97	0.96
	Tenant 2	0.94	0.97	0.95	0.93
Av. assigned capacity ratio	Tenant 1	1.04	1.01	1.01	1.08
	Tenant 2	1.04	1.02	1.08	1.05

4.6.3.3. SLA parameters impact

In this section, the impact of the SLA parameters on achieved behaviour of the proposed approach in the highly unbalanced Situation 4 is discussed.

Fig. 50 compares the temporal evolution of the capacity shares $\sigma_{k,n}(t)$ in cell 2 for Situation 4 from 12 hours to 24 hours against the cell offered load $o_{k,n}(t)$ normalized to the cell capacity c_n. Results are shown for different values of $MCBR_{k,n}$ considering that the same $MCBR_{k,n}$ is configured for all cells and tenants. The rest of the parameters have been configured according to Table XVIII. During the analyzed period, the offered load of Tenant 2 presents very high values between 19 hours until 21 hours, requiring nearly all the capacity in the cell. Instead, the offered load of Tenant 1 remains low, requiring around 35% of the cell capacity. Results show that the trained policies are able to assign the capacity without exceeding the $MCBR_{k,n}$ limit in any case. It is observed that depending on the $MCBR_{k,n}$ value, the assigned capacity shares of the tenants are differently distributed. While for $MCBR_{k,n}$=60% and $MCBR_{k,n}$ =80%, the assigned capacity of Tenant 2 is limited and the offered load of Tenant 1 is satisfied, for $MCBR_{k,n}$=100%, the offered load of Tenant 1 is not satisfied since nearly all the capacity is assigned to Tenant 2. Also, it is observed that for $MCBR_{k,n}$=60% the capacity in the cell is not efficiently assigned since around 15% of the cell capacity remains unused from 18 hours to 23 hours, although this capacity could have been assigned to Tenant 2. These results show the relevance of including the $MCBR_{k,n}$ parameter in the SLA to deal with unbalanced offered loads among cells, contributing to a more efficient and fairer distribution when is adequately configured.

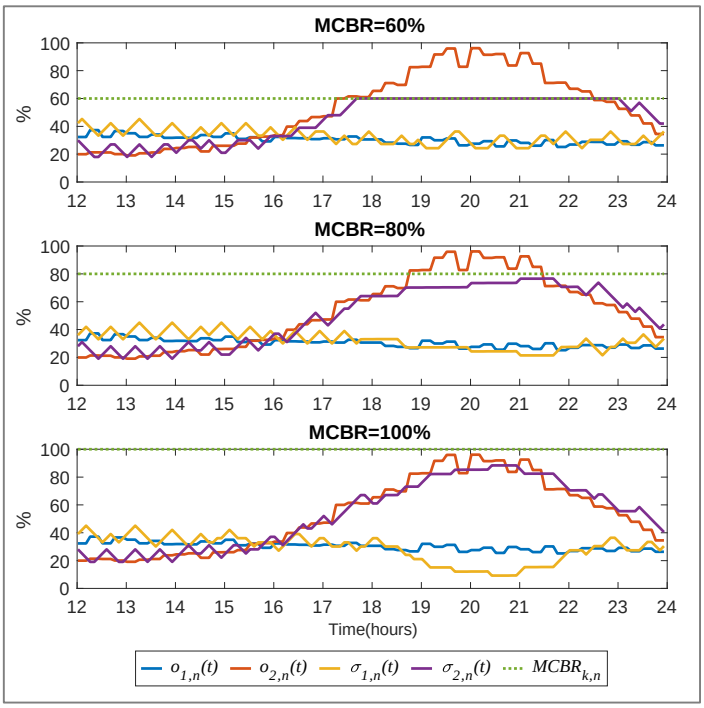


Fig. 50. Capacity shares vs offered load of tenants for cell 2 in Situation 4 and for diverse values of MCBR.

Fig. 51 includes the aggregated offered load among all cells, $O_k(t)$ and the aggregated assigned capacity among all cells, $A_k(t)$, both normalized to the total system capacity C, for both tenants with different values

of $MCBR_{k,n}$ for the same period of Fig. 50. The $SAGBR_k$ values expressed as percentages of the system capacity are also depicted. For a low value of $MCBR_{k,n}$, such as $MCBR_{k,n}$=60%, the aggregated offered loads of both tenants at system level are not satisfied, leading to an inefficient use of the system capacity, since no more than the 60% of the cell capacity can be assigned to any tenant. In turn, when the value of $MCBR_{k,n}$ is increased the aggregated capacity shares $A_k(t)$ of both tenants take closer values to $O_k(t)$. However, $A_k(t)$ still presents lower values than $O_k(t)$, since in some cells there is not enough capacity to satisfy both tenants. Quantitatively, the average system utilization (i.e. the average of the ratio between the summation of $A_k(t)$ of both tenants and the system capacity) for $MCBR_{k,n}$=60% is 77% while for $MCBR_{k,n}$=80% and $MCBR_{k,n}$=100% is 83% and 82%, respectively. Also, good average SLA satisfaction ratios are obtained for both tenants, ranging from 0.9 up to 0.97 for the different analysed $MCBR_{k,n}$ values. The obtained results reinforce the relevance of choosing an appropriate value of $MCBR_{k,n}$ since it also affects the performance at system level.

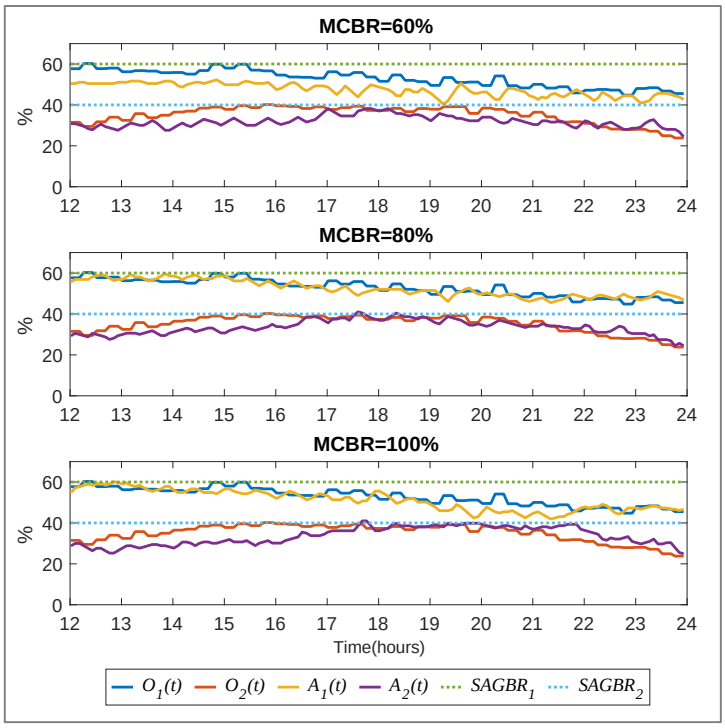


Fig. 51. Aggregated offered loads and assigned capacities in the system.

4.7. Summary

This chapter has presented a DRL-based capacity sharing solution for multi-tenant and multi-cell scenarios. The proposed solution consists in a collaborative MARL approach, where each agent is based on the DQN algorithm. The different DQN agents learn the policy to provide the capacity to be assigned to their associated tenant in each of the cells of the network environment to satisfy the SLA of the tenant and make an efficient use of the available resources. The solution considers that the DQN agents interact with a common network environment in a coordinated and cooperative manner. Moreover, the approach has been contextualised within the 3GPP management framework and has been proposed as a SON function.

The behaviour of the proposed capacity sharing SON function has been assessed in diverse scenarios considering different numbers of cells and temporal and spatial traffic distributions among cells. Results have shown that:

(i) The capacity sharing SON function satisfactorily adapts the capacity assigned to each tenant to their traffic requirements and SLA requirements and achieves making an efficient use of the resources. This is achieved for the different scenarios considered for evaluation, which shows the robustness of the solution to deal with diverse levels of heterogeneity of the traffic distribution among cells.

(ii) The solution has achieved a good trade-off between the two benchmarking capacity sharing schemes considered for comparison achieving the high SLA satisfaction of capacity sharing solutions with hard-slicing (i.e., $SAGBR_k$ is the maximum capacity that can be provided to a tenant) and high resource utilization efficiency of non-slicing solutions (i.e., all the capacity is shared independently of $SAGBR_k$). Moreover, the solution achieves results very close to the optimum when they are applied to diverse offered load patterns.

(iii) The policies learnt by the agents associated to each tenant can be generalizable to any tenant when the dataset used for training is composed of a wide range of traffic requirement situations and SLA requirements.

(iv) The proposed approach is easily scalable to deal with the addition of new tenants simply by associating to the new tenant a new DQN agent with a previously learnt policy and to support the operation when the system capacity changes.

(v) The values of actions and the time step need to be jointly configured to better adapt to the variability characteristics of the traffic requirements of the different tenants in order to improve the SLA satisfaction and maximise the utilisation of the available resources.

(vi) The maximum cell bit rate parameter in the SLA is relevant since it contributes to a more efficient and fair capacity assignment under highly unbalanced traffic situations.

Chapter 5. Implementation aspects of the DRL-based solution for capacity sharing in RAN slicing scenarios

5.1. Introduction

The process that includes the creation, development and operation of Machine Learning (ML)-based solutions, differs from the process used for traditional heuristic and optimisation solutions [96][97]. The main difference is that ML-based logic is not explicitly programmed but rather automatically created based on data. Also, another difference is that multiple feedback cycles between the different stages of the process are required for ML-based solutions, such as model training and model evaluation. Moreover, the process from the model creation until its deployment depends on the ML-type. While the process for supervised an unsupervised ML types rely on the data collection and processing, as described in [96] and [97], Reinforcement Learning (RL) depends on the interaction with an environment, which introduces some differences on the development process of RL solutions.

Fig. 52. presents the workflow that is followed for the modelling, development and production stages, particularized for RL-based solutions, which is described as follows.

Fig. 52. Workflow covering modelling, development and production of RL-based solutions.

The process starts by the modelling stage of the solution. In this stage, the problem under consideration is firstly defined, which embraces the analysis of the problem, the definition of the requirements and goals of the solution, as well as the metrics that will allow evaluating it. After defining the problem, it is assessed which RL algorithm is the most suitable for the solution based on the problem type, action and state spaces dimensions, training duration requirements, the type of interaction with the environment, complexity level, etc. Once a RL algorithm is selected, the RL model can be formulated by particularizing the selected RL algorithm for the specific problem under consideration. This may include the definition of states, actions and rewards or the model scheme (e.g., single agent, multiple agents).

The second stage consists in the model development. The first step of the development process is the coding of the formulated model. There are several libraries available for developing RL-based algorithms, such as *Tensorflow RL Agents* [84], *Keras RL*[98], *OpenAI Baselines*[99] or *Tensorforce* [100], which mainly differ in the programming language, implemented RL algorithms, the quality of the documentation or the possibility to deploy your own environment. The selection of a library should take into account the requirements and features of the defined RL-model. Since the learning in RL is achieved through the trial-and-error interaction of a RL agent with an environment, the environment used for both the training and the evaluation of the developed RL solution needs to be specified. The environment during development will usually be a simulation-based environment that captures the behaviour of the real environment where the solution will be finally deployed. The simulation-based environment can be based on available environments suites, such as those in *Open AI Gym* [101] or *MATLAB* [102], or can be customized by developing them. Moreover, the environments can be supplied with datasets or functions that allow specifying and controlling the states visited by RL agent during the training and evaluation. After this, the developed model is trained in the training environment and the trained model is evaluated in the evaluation environment. This process allows identifying problems in the RL model definition, its coding or its interaction with the environment, which contributes to the model improvement and optimisation. Moreover, as RL solutions include several hyperparameters, these are tuned at this point in order to booster the model performance.

Finally, the production stage consists in the model deployment and the continuous model monitoring. The model deployment includes all those activities that allow operating the trained RL model in the real environment. This may include the integration of the model with the platform where the solution will be executed or the development of the interfaces required for the model to satisfactorily communicate with the real environment. While the trained model operates in the real environment, which is known as inference, the performance of the solution needs to be continuously monitored to detect possible degradations or errors that may arise. Based on this, the model formulation and the developed code can be modified or the solution can be re-trained.

Considering the above stages, this chapter discusses on diverse relevant implementation aspects for the deployment of DRL-based solutions that support the management of RAN slicing in real networks, taking as a reference the proposed DRL-based capacity sharing solution in the previous chapter. First, given that the implementation of RL-based solutions for RAN management must comply with the standards, Section 5.2 provides an overview on the standardization work held by 3GPP and O-RAN and proposes a compatible framework for the integration of the proposed DRL-based capacity sharing solution with their standards. Second, Section 5.3 discusses on the training stage of RL-based solutions for capacity sharing and conducts an impact analysis of the features of the data used for training on the performance during both the training and inference stages by considering the DRL-based capacity sharing solution in the previous chapter. Section 5.4 details the development and production stages for the particular case of the proposed DRL-based capacity sharing solution, which covers the description of the tools and technologies required to

develop the solution and the interfaces according to the specified technologies by O-RAN. Finally, Section 5.5 summarizes the main conclusions extracted from this chapter.

5.2. Compatibility of DRL-based capacity sharing solutions for RAN slicing scenarios with 3GPP and O-RAN standards.

A relevant aspect for the application of DRL-based solutions for RAN management in real network environments is its compatibility with current standardization work for 5G networks. On the one hand, the O-RAN alliance is working on the standardisation of an open architecture that will support the virtualisation of the RAN and the usage of ML-based solutions for the RAN at different levels. Therefore, the integration and compliance of the DRL-based solutions for RAN slicing management with the standardized architecture by O-RAN provide the solutions with a robust framework to train and deploy them. On the other hand, the 3GPP has included a set of information models, which represent SLAs, network instances' features and slicing-related configuration parameters of 5G base stations, as well as management services for the management of the lifecycle of network slices. Also, 3GPP provides performance management (PM) services that allow the collection of performance measurements and KPIs, required for the monitoring of the network. The compliance of DRL-based solutions for RAN slicing management with the 3GPP information models and PM services provides them with standardized input and outputs to be integrated in real deployments.

This section firstly provides an overview of the main elements of O-RAN architecture and the application of ML-based assisted solutions in the architecture in Section 5.2.1. Then, the 3GPP information models and management services related to capacity sharing in RAN slicing scenarios are described in Section 5.2.2. Finally, a framework for the integration of the DRL-based capacity sharing solution proposed in the previous chapter with the 3GPP information models and management services as well as O-RAN architecture and procedures is provided in Section 5.2.3.

5.2.1. O-RAN architecture overview

O-RAN Alliance is working on the development of a more open, smarter, interoperable and scalable RAN. The O-RAN architecture aims at being the foundation for building a virtualized RAN on open hardware and cloud with embedded AI, which will empower network intelligence through open and standardised interfaces in multi-vendor networks [85]. This architecture is based on O-RAN Alliance standards, which are supported and complementary to 3GPP standards and other industry standard organizations.

The O-RAN logical architecture is depicted in Fig. 53, which shows the different O-RAN functions and interfaces as well as 3GPP interfaces integrated in the architecture [86]. The Service Management and Orchestration (SMO) function is responsible for the management of the rest of O-RAN functions and the O-Cloud. These are the O-RAN Central Unit – Control Plane (O-CU-CP), O-RAN Central Unit – User Plane (O-CU-UP), the O-RAN Distributed Unit (O-DU), the O-RAN Radio Unit (O-RU) and the O-RAN eNB (O-eNB), which can be deployed whether as Virtualized Network Functions (VNFs), such as virtual machines or containers, or as Physical NF (PNFs) utilizing customized hardware. Four key interfaces are

introduced, namely A1, O1, Open Fronthaul (FH) M-plane and O2, that connect the SMO with the rest of O-RAN functions and the O-Cloud. To support AI in the framework, the near-real-time RAN Intelligent Controller (near-RT RIC) function is included that is in charge of the control of the radio protocol of the stack processing within the O-CU-CP/UP and O-DU via the E2 interface, providing enhanced AI-powered RRM functionalities. Also, the SMO contains the non-real-time RAN Intelligent Controller (non-RT RIC) function, responsible for the policy management, RAN analytics and ML model management and delivery of enriched information for near-RT RIC operation through the A1 interface. For further details on O-RAN functions and interfaces, the reader is referred to [86].

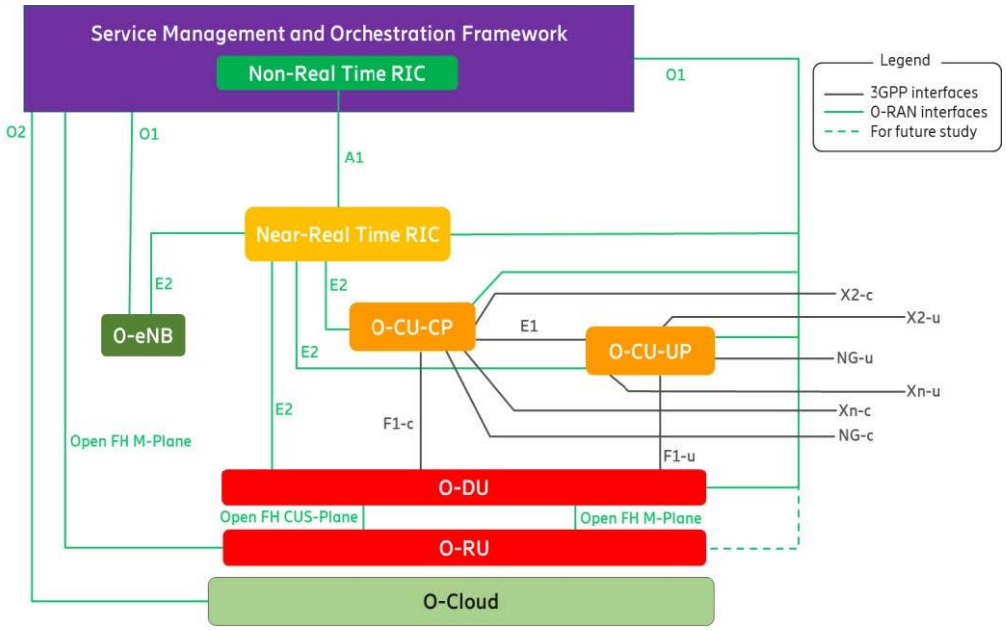


Fig. 53. O-RAN Logical Architecture [86].

Building upon the O-RAN architecture, the O-RAN alliance is working on the definition of an open framework for the applicability of ML-assisted solutions [87], which defines the required AI/ML components as well as procedures and interfaces for the ML-assisted solutions lifecycle. Fig. 54 shows a simplified model of the main components and relations under O-RAN for the training and inference of ML-assisted solutions. The ML training host is a network function which hosts the training of the model, including offline and online training based on data collected from the RIC, the O-DU or the O-RU. Instead, the ML inference host hosts the ML model during inference mode (which includes both the model execution as well as any online learning if applicable) and informs the actor about the output of the ML algorithm. The actor takes the decision based on output provided by the ML inference host. The actor and the ML-Inference host often coincide. The location of the elements in Fig. 54 in the O-RAN Architecture of Fig. 53 depends on the ML solution and solution types. In the case of RL-based solutions, diverse options are devised to place the different components (i.e., ML training host, ML Inference Host and Actor) in the non-RT-RIC of SMO or Near-RT-RIC, which depend on the training needs of the solution i.e., online training, offline training, separated training and inference, etc.

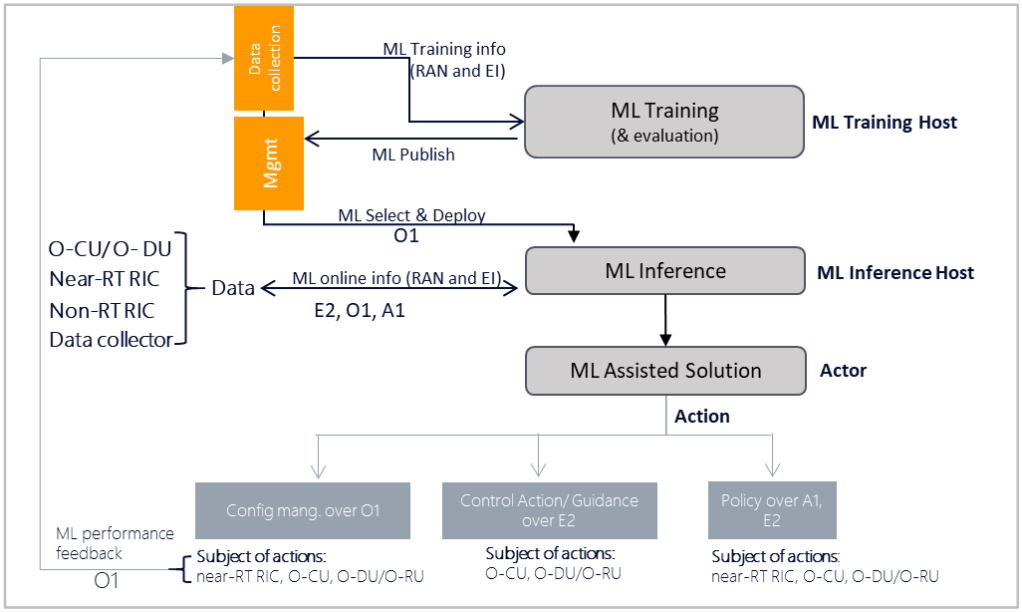


Fig. 54. ML components and relations for ML-assisted solutions within O-RAN [87].

5.2.2. 3GPP vision on capacity sharing for RAN slicing

Support for network slicing has been firstly introduced by 3GPP as part of the first release of the Fifth Generation (5G) system specifications (Release 15), with multiple enhancements included in Releases 16-18 [88]-[92].

The problem of capacity sharing treated by the DRL-based capacity sharing solution described in Chapter 4 has been identified by 3GPP as a use case in the context of Self-Organizing Network (SON) feasibility studies [81], referred to as “cross-slice network resource utilization”. This SON service supports the management of network slices by adjusting the policies and resource adjustment according to the requirements of these slices, not only considering the dynamic allocation of radio resources to slices but also the distribution of other resources such as storage or computing resources.

Regarding the management of the network slices, 3GPP provides specifications on the management of network slicing, including information model definitions, referred to as Network Resource Models (NRMs), for the characterization of network slices [52] together with a set of management services (MnS) for network slice lifecycle management (e.g. network slice provisioning MnS for network slice creation, modification and termination, performance monitoring services per slice, etc.) [16].

3GPP information model includes definitions for “network slice NRM” [52], that include two information object classes (IOC), named *NetworkSlice* and *NetworkSliceSubnet*, to represent the managerial representation of the network slice within the Network Slice Provider (NSP) OSS. The *NetworkSlice* IOC and the *NetworkSliceSubnet* IOC represent, respectively, the properties of a network slice instance (NSI) and a network slice subnet instance (NSSI) in a 5G network. Note that the realization of a NSI may be tied to the realization of several NSSIs (e.g. a NSI composed of RAN NSSI and a 5G Core NSSI) or only a

single domain (e.g., NSI consisting of a single RAN-only NSSI). Within the *NetworkSlice* IOC and *NetworkSubnetSlice* IOC, the set of attributes that define the requirements to be supported by the NSI and the NSSI slice requirements are named, respectively, *ServiceProfile* and *SliceProfile*. The attributes included in the include *ServiceProfile* and *SliceProfile* can be directly inherited from the standardized list for network slices provided by the GSMA GST [93] as well as additional attributes to capture more specific requirements derived from the service performance requirements defined in [94][95]. For the case of the DRL-based capacity sharing solution in Chapter 4, let us introduce here three attributes included in the *ServiceProfile* of the solution:

- *dlThptPerSlice*: It defines the achievable aggregate downlink data rate of the network slice.
- *dlThptPerUe*: It defines the average data rate delivered by the network slice per UE.
- *termDensity*: It specifies the maximum user density over the coverage area of the network slice.

Moreover, 3GPP also provides information model definitions for "New Radio NRM" [52], where different RAN management parameters are defined to configure the behavior of the RAN nodes with regard to the operation of the established network slices. In this respect, Fig. 55 provides a simplified view of the classes and attributes of the "New Radio NRM" model for the characterization of the RRM policies for the resource allocation to the slices. On the top, the *RRMPolicyManagedEntity* proxy class represents the different RAN managed components (e.g., cell resources managed at DU, cell resources managed at CU functions, DU functions, etc.) that are subject to the RRM policies. Then, the *RRMPolicy_* IOC represents the properties of an abstract RRMPolicy that defines two attributes: the *resourceType* attribute, used to define the type of resource (e.g. PRB, RRC connected users, etc.) and the *rRMPolicyMemberList* attribute, used to indicate the associated network slice or group of network slices that is subject to this policy. The associated network slices are specified here in terms of slice identifiers such as the Single Network Slice Selection Assistance Information (S-NSSAI) and Public Land Mobile Network Identifier (PLMNid).

On this basis, the *RRMPolicyRatio* IOC represents a particular realization of an RRMPolicy that establishes a resource model for resource distribution among slices based on three resource categories: shared resources (shared among slices with no specific guarantees per slice), prioritized resources (guaranteed for use by associated slices but still usable for other slices when free) and dedicated resources (only used for the associated slices). Accordingly, the following attributes are included in the *RRMPolicyRatio* IOC:

- *rRMPolicyDedicatedRatio*, defines the dedicated resource usage quota for the associated network slice(s), including dedicated resources. The sum of the *rRMPolicyDedicatedRatio* values shall be less or equal than 100.
- *rRMPolicyMinRatio*, defines the minimum resource usage quota for the associated network slice(s), including prioritized resources and dedicated resources, which means the resources quota that need to be guaranteed for use. The sum of the *rRMPolicyMinRatio* values shall be less or equal than 100.

- *rRMPolicyMaxRatio*, defines the maximum resource usage quota for the associated network slice(s), including shared resources, prioritized resources and dedicated resources. The sum of the *rRMPolicyMaxRatio* values can be greater than 100.

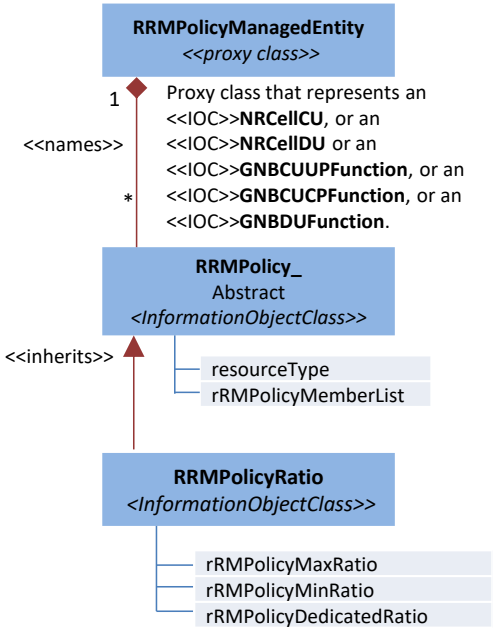


Fig. 55. Simplified representation of the classes and attributes for configuration of RAN functions subject to RRM policies for slice management.

Moreover, 3GPP provides specification for the performance assurance-related management services for 5G networks, including network slicing, that allow the collection of real-time performance measurements that can be consumed by analytic applications such as SON to take appropriate actions [82]. The provisioning of performance measurements can be reported per NSSI or per NF based on two main services:

- *Performance data file reporting service*, which enables an authorized PM consumer to get a performance data file of a NSSI/NF. The PM consumer is notified with a *notifyFileReady* notification when a performance data file is available in order to download it by using FTP or SFTP.

- *Performance data streaming service*, which enables an authorized PM consumer to receive the performance data stream of NSSI/NF.

To support these Performance Management (PM) services, 3GPP specifies performance measurements for NG-RAN including network slicing in [41], and based on these, end-to-end Key Performance indicators (KPIs) in [103]. For the case of the DRL-based capacity sharing solution in Chapter 4, the following PM measurements and KPIs are considered:

- "DL PRB used for data traffic", which measures the number of PRBs used in average for data traffic in a given slice and cell. This performance measurement is defined in [41].

- "DL total available PRB", which measures the number of available PRBs in the cell. This performance measurement is defined in [41].

- "Downstream throughput for Single Network Slice Instance", which captures the aggregated throughput per slice across all the cells. This KPI is defined in [103].

5.2.3. Compliance of the DRL-based capacity sharing solution for RAN slicing scenarios with 3GPP/O-RAN framework.

Considering the above, this section describes the compatibility of the DRL-based capacity sharing solution proposed in Chapter 4 with the O-RAN and 3GPP standards. The 3GPP/O-RAN compatible framework of the DRL-based capacity sharing solution is illustrated in Fig. 56. The DRL-based capacity sharing solution is conceived to be deployed as part of the RAN SMO as a SON function for cross-slice resource optimization solution. The DQN-based agents of the MARL approach are trained on the ML training host based on real or synthetic data in the network. Once trained, the policies associated to each slice are deployed in the ML inference host, which is co-located with the actor, where the DRL-based capacity sharing inference module takes place.

The DRL-based capacity sharing inference makes dynamic decisions on the configuration of *rRMPolicyDedicatedRatio* attribute of each cell on a per RAN slice basis every Δt minutes. This configuration is conducted via the management provisioning services offered by the O1 interface. In particular, since the *rRMPolicyDedicatedRatio* attribute establishes the resource usage quota assigned to the RAN slice defined in terms of the fraction of Physical Resource Blocks (PRBs) that can be used by this slice, the attribute is configured in the O-DU unit, so that the MAC layer can take this resource usage quota into account when allocating PRBs to the users of the RAN slice in the cell. The *rRMPolicyDedicatedRatio* is mapped with the capacity share $\sigma_{k,n}(t)$ of the DRL-based capacity sharing solution in Chapter 4. To make the configuration on a per-slice basis, an *rRMPolicyMemberList* is specified for each RAN slice, being composed of a single member with the S-NSSAI and PLMNid of the RAN slice. Then, the *rRMPolicyDedicatedRatio* is configured per *rRMPolicyMemberList* in each cell.

Moreover, the requirements of each RAN slice are specified by means of the three *ServiceProfile* parameters explained in Section 5.2.2, namely *dlThptPerSlice*, *dlThptPerUe* and *termDensity*. These parameters are related to the SLA parameters established for each tenant in the DRL-based capacity sharing solution. The $SAGBR_k$ corresponds to the *dlThptPerSlice* whereas the $MCBR_{k,n}$ is obtained by the product of the *dlThptPerUe*, the *termDensity* and the cell *n* service area.

The decisions on the *rRMPolicyDedicatedRatio* for each RAN slice are taken based on the state. According to the definition of the state of the DRL-based capacity sharing solution in Section 4.4.1, the state relies on the parameters in the *ServiceProfile* and the *rRMPolicyDedicatedRatio* for each of the cells, which are available by the DRL-based capacity sharing inference module. Moreover, the state requires of performance measurements for the computation of the resource usage $\rho_{k,n}(t)$ and the fraction of PRB not used by any tenant in the cell $\rho_n^A(t)$. The $\rho_{k,n}(t)$ and $\rho_n^A(t)$ correspond to the the "DL PRB used for data traffic" and the "DL total available PRB" performance measurements defined in [41], respectively, both normalized to the

total number of PRBs in the cell W_n. Based on the state, the policy associated to the different RAN slice are executed and then, the resource usage quota computation function computes the *rRMPolicyDedicatedRatio* of each RAN slice in each cell, according to the procedure described in Section 4.4.5, which ensures the $MCBR_{k,n}$ is not exceeded and that the aggregated resource usage quota for all the slices in a cell after applying the actions does not exceed 1 in order not to exceed the cell capacity. This function relies on the aggregated throughput experienced by the slice across all cells, $T_k(t)$, which can be measured by the KPI "Downstream throughput for Single Network Slice Instance" defined in [103]. Note that the performance measurements and KPIs can be obtained by consuming PM services, whether by using a *Performance data file reporting service* or a *Performance data streaming service* depending on the duration of Δt.

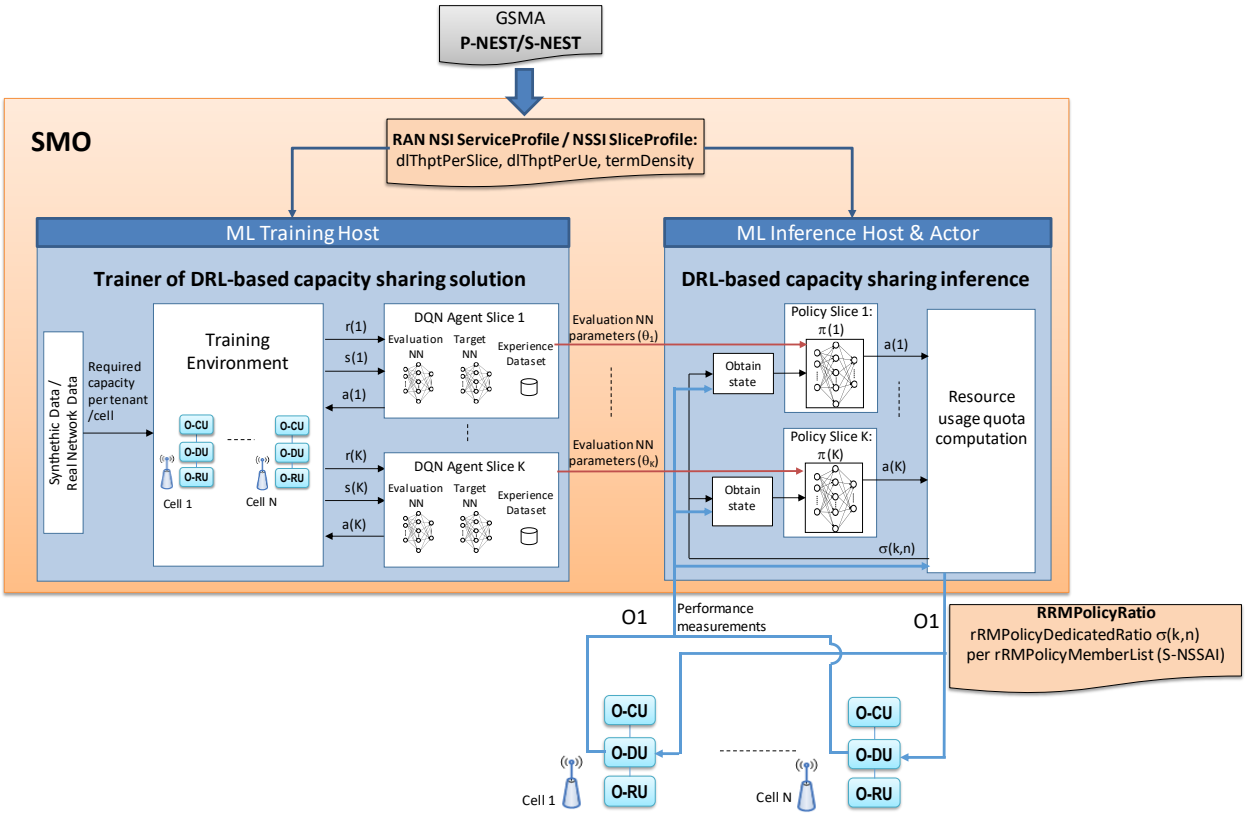


Fig. 56. 3GPP/ORAN compatibility of DRL-based capacity sharing solution.

Overall, the framework in Fig. 56 presents a plausible integration of the DRL-based capacity sharing solution with the 3GPP information models and management services as well as O-RAN architecture and procedures. This is essential for the application of the solution in real network environments given that equipments and platforms that are 3GPP-compliant and O-RAN standard are expected to be adopted for the RAN virtualization and the integration of ML-based solutions.

5.3. Training of DRL-based capacity sharing solutions for RAN slicing scenarios

Although a solid background has been generated on methods and algorithmic solutions for DRL-based capacity sharing solutions [69]-[79], the detailed analysis of the state-of-the-art allows identifying a key research gap, namely, how the training stage of DRL is conducted and how training is distinguished from

the inference stage (i.e., the execution of a trained policy on the real network). Indeed, previous DRL-based solutions for capacity sharing in [69]-[77] have not explicitly distinguished the training and inference stages. Only the works in [78] and [79] make this distinction and discuss on the generalization capability of the learnt policies when doing the training with a large number of situations. However, none of the above works has provided insights into the features and requirements of the data used for training and their impact on the achieved performance.

This section fills this gap by performing a quantitative analysis of the impact of training on the performance of the policies learned by DRL-based capacity sharing solutions. To this end, the features that characterise training datasets used for DRL-based capacity sharing solutions are identified and, then, a methodology for building training datasets with given features is introduced in Section 5.3.1 that allows properly configuring the training environment. In addition, several metrics to quantify the impact of the built training datasets are defined. Afterwards, results assessing the methodology when applied to the DRL-based capacity sharing solution proposed in Chapter 4 are provided in Section 5.3.2.

5.3.1. Methodology for training impact analysis

The intrinsic online learning behaviour of DRL becomes a major challenge for its application in the capacity sharing problem. Due to the high impact of capacity sharing decisions on the network performance and efficiency, erroneous actions selected by DRL agents during the trial-and-error learning process may lead to unacceptable performance degradations in the real network. Also, since capacity sharing solutions operate at time scales in the order of seconds to minutes, long training durations would be required for the DRL agents to experience a sufficient variety of situations in the real network. To tackle this problem, it is possible to train the DRL agent in a simulated training environment that emulates the behaviour of the real network. The training environment is configured by means of a training dataset that determines the situations that the DRL agent will experience during the training. In this way, the agent can accumulate sufficient experience in a wide variety of situations in order to learn the decision-making policy that will be later on applied on the real network during the so-called inference stage.

An adequate training needs to ensure that a sufficiently representative number of situations are encountered by the DRL agent during the training stage, so that it can properly generalize from them in order to make the best decisions during the inference stage. In the case of capacity sharing, since DRL agents decide on the capacity assigned to each tenant according to their traffic needs during time, the configuration of the training environment should be based on the time evolution of the traffic demand (i.e., offered load) of each tenant. In this thesis, we model this evolution as a temporal pattern of one-day duration. Then, a training dataset is defined as a set of temporal patterns that will be used as inputs to the training environment and experienced by the DRL agents during the training stage. The way to generate this dataset and the metrics for analysing the impact of training is described in the following.

5.3.1.1. Training stage

Let us consider the availability of J temporal patterns, where each pattern j represents the time evolution of the offered load of a tenant during a day, $O_j(t)$, in a cell. The offered load is the required capacity in bits/s of a tenant normalized to the total cell capacity, so that it falls in the range [0,1]. It is obtained for t=0…T in time steps of duration Δt. Offered load temporal patterns can be based on historic real data extracted from the real network or can be generated synthetically.

Every temporal pattern is characterized in terms of two features: (1) Dynamic Range (D), which is the difference between the maximum and the minimum values of $O_j(t)$; (2) Level (L), which corresponds to the average value of $O_j(t)$. For both features, the temporal patterns are classified into *High* and *Low*, leading to the four categories depicted in Fig. 57, where each one is given by a combination of classes of D and L, e.g., a category is {D=*High*, L=*High*}. The number of patterns in each category should be sufficiently represented. In the case that the available temporal patterns are not enough, which may occur e.g., when scarce real data measurements are used, the number of patterns can be augmented by adding a random component (e.g., uniformly distributed) to each of the available temporal patterns, emulating the traffic fluctuations.

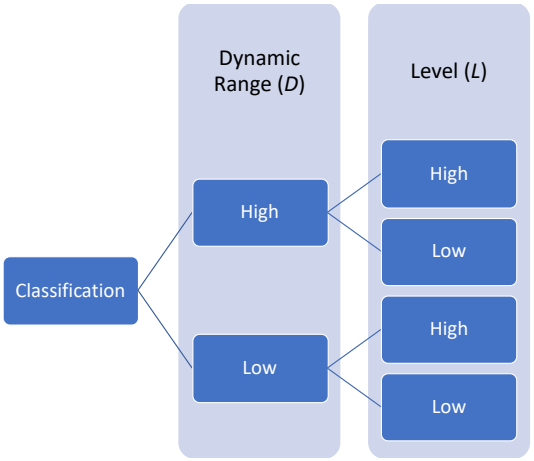


Fig. 57. Temporal patterns classification.

The available J temporal patterns are used to build the training dataset. Since capacity assignment decisions for a tenant depend on both the offered load of that tenant and the offered loads of the rest of the tenants, the training of a DRL agent considers the offered load of a target tenant, denoted as tenant 1, and the aggregate offered load of the rest of the tenants, which for simplicity can be referred here to as tenant 2. Hence, the training dataset is composed of a total of Q pairs of one temporal pattern for tenant 1 and another one for tenant 2, which are selected from the set of J patterns. The Q pairs are randomly ordered to generate the time evolution of the offered load that is injected to the simulated environment during the training process.

The features of the selected patterns for each tenant determine the actual offered load values of tenant 1, o_1, and tenant 2, o_2, that are included in the training dataset. Then, to assess the range of offered load values in the training dataset, the *coverage* of a dataset, *COV*, is defined. It is the ratio between the number of offered

loads (o_1,o_2) included in the training dataset over the space of possible load values, which considers the load in the range [0,1] and discretized in steps of Δo. A training dataset with *COV* value close to 100% will allow that the DRL agent experiences almost all the possible offered load pairs of both tenants during the training stage.

During the training stage, the DRL agent will be making decisions in accordance with the experienced offered load values included in the dataset and progressively updating the decision-making policy based on the outcome of these decisions. This process is executed until the DRL agent achieves convergence in the learnt policy. To define the convergence condition, the well-known loss metric (see definition in equation (58)) is considered. The loss tends to decrease and stabilize (i.e., loss fluctuations are small) when the DRL agent successfully learns a good approximation of the optimum policy, and thus, convergence is achieved. To quantify convergence, the loss is averaged in periods of M training time steps. Then, the loss variation $v(m,W)$ at period m over the window of the last W periods is defined as the difference between the maximum and the minimum average loss during the last W periods divided by the total average loss over all the W periods. Based on this, the criterion considered here is that convergence is achieved at period m when $v(m,W)$, is lower than a given threshold v_{th}. The number of time steps until reaching this condition determines the convergence time, t_c. Moreover, to assess the training stability, the standard deviation of the average loss in the window $[t,t+W_S)$, denoted as $\sigma(t,W_S)$, is also obtained.

5.3.1.2. Inference stage

After the training is completed, the policy can be applied in the real network environment to assign the capacity to the tenants according to their actual experienced offered loads. The achieved performance of the policy during inference depends on the generalization capability of the trained policy to adapt to offered load situations not included in the training dataset but also on the similarity between the offered loads included in the training dataset and those experienced during inference. To measure this similarity, the *percentage of coincidence*, *PC*, is defined as the percentage of the observed offered load values (o_1, o_2) during inference that had been included in the training dataset more than d_{min} times. For illustrative purposes, Fig. 58 shows the offered loads experienced during an inference stage, represented as bullets, on top of a training dataset 2D-histogram. The training dataset contains patterns that belong to the category {*D=Low*, *L=Low*} for both tenants and presents a *COV*=22.6%. The offered load values experienced during inference present a high similarity to the ones included in the training dataset, achieving a *PC*=95% when d_{min}=100.

During the inference stage, the performance of the trained policy is also assessed by obtaining the aggregated reward of both tenants at each time step t, $R(t)$, which rates how good were the capacity allocation decisions for tenant 1 and tenant 2 for the actual experienced loads.

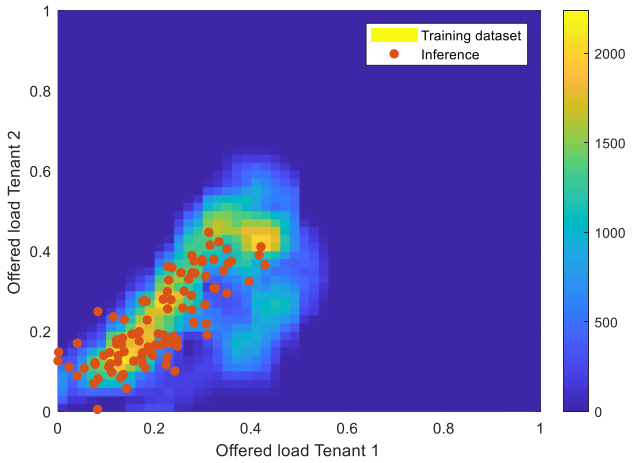


Fig. 58. Training dataset vs inference stage experienced load.

5.3.2. Impact analysis results

5.3.2.1. Considered scenario and datasets

The DRL-based capacity sharing solution proposed in Chapter 4 has been considered for studying the impact of training. The scenario considered for evaluation is comprised of a single cell, which serves the users of K=2 tenants, denoted as *Tenant* 1 and *Tenant* 2. The scenario considers that the cell has a capacity c_n=117 Mb/s and the established SLAs are $SAGBR_1$=70.2 Mb/s and $SAGBR_2$=46.8 Mb/s, corresponding to the 60% and the 40% of the cell capacity, respectively, and $MCBR_1$= $MCBR_2$=93.6 Mb/s, corresponding to the 80% of c_n. The DQN parameters considered for training are a time step duration Δt = 3 min and an action step Δ=0.03. The rest of DQN parameters used for training are those of Table XIX.

To build the training datasets, a total of J=8400 offered load temporal patterns of one-day duration have been considered and classified according to the dynamic range D and offered load level L. For illustrative purposes, Fig. 59 includes two offered load temporal patterns considered in the dataset that present different features: the offered load temporal pattern j=10 has a low dynamic range and low offered load level while the offered load temporal pattern j=4 presents a high dynamic range and achieves high offered load levels. Moreover, Fig. 60 depicts a representative set of the available temporal patterns, the central mark on each box indicating the median of temporal pattern, the bottom and top edges of the box indicating the 25th and 75th percentiles, respectively, and the whiskers extending to the most extreme data points. The representation of Fig. 60 shows that temporal patterns with diverse dynamic ranges and offered load levels are included in the dataset. For instance, the pattern j=18 belongs to the category {*D=High*, *L=Low*} and the pattern j=9 to {*D=Low*, *L=Low*}. The rest of patterns have been obtained from those of Fig. 60 by adding a random component uniformly distributed in the range [-0.05,0.05] until having a total of 2100 patterns for each of the possible 4 categories.

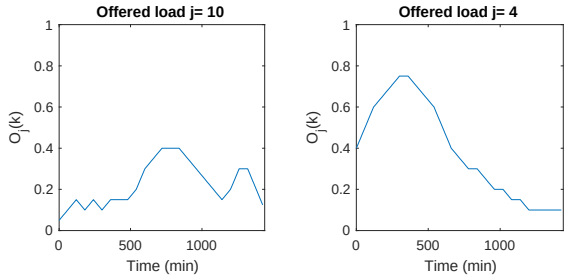


Fig. 59. Offered load temporal patterns *j*=10 and *j*=4.

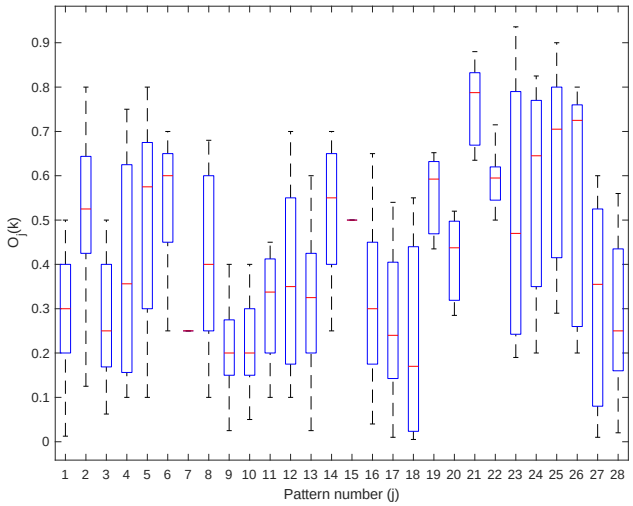


Fig. 60. Temporal patterns analysis.

5.3.2.2. Impact of the training dataset on the training stage

This section studies the impact of the features of training datasets on the training process. To this end, the training of the DRL-based capacity sharing solution has been performed for two datasets, denoted as *Dataset* 1 and *Dataset* 2, both of them with Q=4200 pairs of temporal patterns of each tenant. *Dataset* 1 is a specific dataset that includes temporal patterns that only belong to the category {*D=Low*, *L=Low*} for both tenants. Instead, *Dataset* 2 consists in a generic dataset that includes temporal patterns belonging to all the categories. The coverages of *Datasets* 1 and 2 are *COV*=22.57% and *COV*=92%, respectively. The training of both datasets has been performed 50 times, each time with a different order of the pairs of patterns included in the datasets. This allows mitigating potential effects that the random order in which the offered loads are experienced by the DQN agent and the randomness of the ε-Greedy policy, which selects a random action with probability ε, may have on the training evolution.

Table XXVI includes the 95th percentile of the convergence time t_c and the average and 95th percentile of the standard deviation of loss $\sigma(t,W_S)$ over the different executions of the training for datasets 1 and 2. The computations assume a loss average period M=5000 time steps, a variation window W=10·10^4 time steps, a convergence time threshold v_{th}=0.06 and a standard deviation window W_S=50·10^4 time steps. Results indicate that convergence is faster with *Dataset* 1 than with *Dataset* 2, being the 95th percentile of the convergence time with *Dataset* 1 approximately 25% less than with *Dataset* 2. The reason is that *Dataset* 1, which only includes patterns from a specific category, covers a lower number of states than *Dataset* 2,

so a shorter number of iterations are required to go through all of them. It is also observed that the standard deviation of the loss is also smaller when using *Dataset* 1 than when using the *Dataset* 2. Both the average and the 95th percentile obtained with *Dataset* 2 are approximately twice than those with *Dataset* 1, which reflects that a much more stable learning is achieved for *Dataset* 1 due to its smaller coverage. Therefore, it is concluded that the coverage of a dataset, and thus, its level of specificity or generality, has a direct impact on the process of training, achieving faster convergence and higher stability those datasets with low *COV*.

TABLE XXVI. PERFORMANCE OF THE TRAINING STAGE.

Performance Metrics	Dataset 1	Dataset 2
95th percentile of time to converge t_c (time steps)	$75 \cdot 10^4$	$105 \cdot 10^4$
Average standard deviation of loss $\sigma(t,W_S)$	$9.25 \cdot 10^{-4}$	$21 \cdot 10^{-4}$
95th percentile of standard deviation of loss $\sigma(t,W_S)$	$20 \cdot 10^{-4}$	$56 \cdot 10^{-4}$

5.3.2.3. Impact of the training dataset on the inference stage

In this section, an impact analysis of the features of the training dataset on the learnt policy is conducted. To perform this analysis considering a variety of coverage values, 16 training datasets have been defined, exhibiting levels of *COV* ranging from as low as 0.04% to as high as 92%. Then, the policy learnt with each training dataset has been analysed during the inference stage. For the inference stage, a simulator of the real network has been considered that takes as input 80 different pairs of temporal patterns for tenant 1 and tenant 2 that belong to a wide range of dynamic ranges and offered load levels.

In order to assess the performance of a policy during the inference stage, the assigned capacity share by the policy in each time step is compared to the optimum value, obtained by means of an exhaustive search algorithm. Specifically, in each time step the aggregate reward of both tenants is computed for all the possible values of capacity shares of Tenant 1 and Tenant 2, discretized in steps of Δ. Then, the one that achieves the maximum aggregate reward of both tenants is selected as the optimum. Based on this, the optimality ratio, defined as the ratio between the aggregate instant reward $R(t)$ obtained by the learnt policy and the optimum aggregate instant reward, is computed in each time step. The optimality ratio is recorded for each pair of offered loads of tenant 1 and 2 (discretized in steps of Δo=0.02) observed during the inference stage. Then, we define the metric denoted as $P_{0.9}$ as the percentage of offered load pairs with optimality ratio higher than 0.9.

Fig. 61 shows the value of $P_{0.9}$ for each learnt policy as a function of the coverage *COV* of the training dataset used to obtain the policy. The trend line of Fig. 61 reveals that, for *COV* smaller than approximately 25%, $P_{0.9}$ grows when increasing *COV*, while for *COV* higher than 25%, it flattens at around 80%. This reflects the strong generalization capability of the considered DQN approach since with a training dataset of relatively low *COV* value such as 25%, it achieves a performance close to the optimum for most of the offered load levels in the inference stage. In other words, for *COV* above 25%, during the execution of the

DQN-based algorithm in a practical scenario, appropriate decisions will be taken even though the actual offered loads levels have not been observed during the training stage.

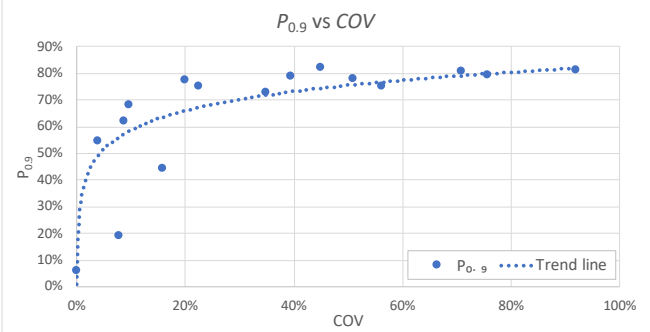


Fig. 61. $P_{0.9}$ with respect to *COV*.

Fig. 61 also shows that the performance in terms of $P_{0.9}$ for datasets with low *COV* is much worse, and suffers strong variations depending on the actual offered loads that were included in the training dataset. In this case, the percentage of coincidence *PC* metric between the offered loads in the inference stage and the training datasets becomes especially relevant for the analysis. To show this, the learnt policy for a training dataset with *COV*=8% has been assessed for the case of experiencing a *PC*=100% and *PC*=32% during the inference stage, considering d_{min}=100 in the computation of *PC*. It is obtained that for the case of *PC*=100%, $P_{0.9}$=94% is achieved whereas for the case of *PC*=32%, only $P_{0.9}$=48% is obtained. This proves that the achieved performance for datasets with low *COV* has high dependency on the *PC*.

5.4. Development and production stages of DRL-based capacity sharing solution for RAN slicing scenarios

Between the definition of a RL-based solution and its application in a real production environment, the solution needs to go through different stages to be successfully implemented. After the modelling of the RL-based solution at algorithmic level, the development stage follows, which aims at providing a validated and robust software solution. Next, the production stage takes place, which allows the satisfactorily deployment and operation of the developed software solution in the production environment.

Relate to the above, this section describes the development and production stages of the DRL-based capacity sharing solution proposed in Chapter 4. Specifically, Section 5.4.1 introduces the main software tools that have been employed in the DRL-based capacity sharing solution' development and provides a high-level overview of the developed solution. Afterwards, Section 5.4.2 describes the production stage of the solution, which includes an overview of the technologies required for the deployment of the solution in a production environment as well as the description of the adopted solution for the production stage.

5.4.1. Development phase of DRL-based capacity sharing solution for RAN slicing scenarios

In this section, the development stage of the DRL-based capacity sharing solution modelled in Chapter 4 is described. The model has been developed in Python by using the library *Tensorflow RL Agents* [84], also known as *TF-Agents*. This library has been selected because it provides open-source implementations of most of the current state of the art DRL algorithms and it allows defining multi-agent schemes and

customized environments, as required for the model development. Also, the library includes several tutorials and documentation as well as an extensive community, which ease the development and customization of the different elements of the solution.

Fig. 62 shows the system overview of the *TF-Agents* system. The system separates the collection of experiences from the training of the agents. The collection of experiences is performed in an environment, which can be based on available environment suites (e.g., Open AI Gym, Atari) or a self-customized environment developed in Python. Also, the environment defines the possible states and actions and, in each time step, computes and stores the reward and new state for a selected action. The system allows the definition of multiple environments in parallel, where the data collection is controlled by the driver. The driver also gathers the different trajectories (i.e, experiences), which are collected in the replay buffer (i.e., experience replay). Then, the training stage is performed by batching the data in the replay buffer and using it to update the agent's model, e.g., the deep neural network in DQN. Based on this, the updated policy is collected by the driver to select the actions sent to the environment for data collection. The policy can be collected as an ε-Greedy policy to conduct the training or as a Greedy policy to be used for the inference.

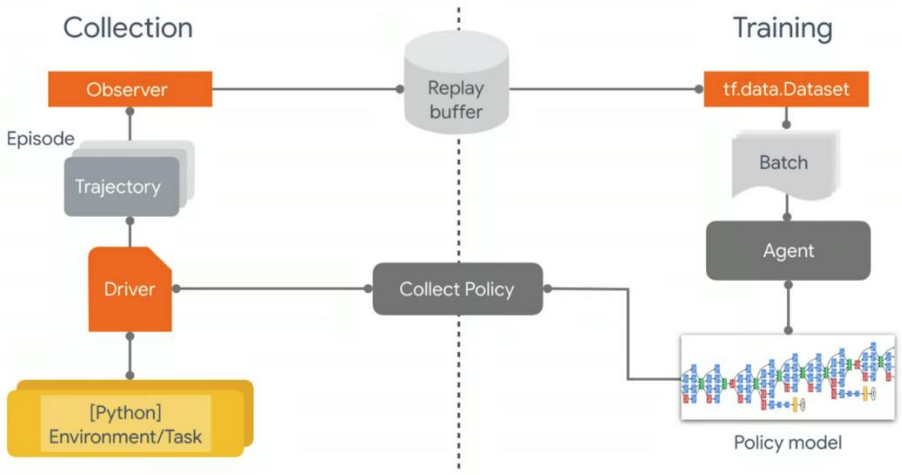


Fig. 62. TF-Agents system overview [104].

Based on this system, Fig. 63 shows the developed solution for the training of the DRL-based capacity sharing solution by using the DQN implementation of TF-agents library. Specifically, the solution includes a DQN Agent for each tenant in the scenario. To support the DQN Agent of each tenant, a separate experience replay and Tensor Flow environment, referred to TF-environment, is created. This is required since each TF-Environment allows the interaction with the DQN Agent and contains the states and rewards for each of the tenants. Each DQN Agent is trained independently based on the experiences in its own experience replay. To manage the different agents, a *Driver* is included that collects the policies of the different agents in addition to training metrics (e.g., loss). The driver uses the collected policies to trigger the actions according to the state in the TF-Environment of each agent in a per-time step basis. The different TF-Environments interact with a common Python environment that has been self-developed and simulates a network deployment. It is composed of a scenario controller and *N* cells. The scenario controller has two

main tasks. First, it communicates with the different TF-Environments, receiving the actions that configure the capacity sharing parameters in the scenario and providing each TF-Environment with the states and rewards as a result of their actions. Second, it acts as a manager of the different cells, configuring them according to the actions provided by the TF-Environments and gathering the required data to compute the state and rewards provided to the TF-Environments. The cells include all the operations required to compute the tenant and cell level performance according to the capacity sharing configuration at each time step and the traffic demands in the cell by each tenant. In addition, the Python environment includes a module for reporting and monitoring the KPI evolution at system and cell levels.

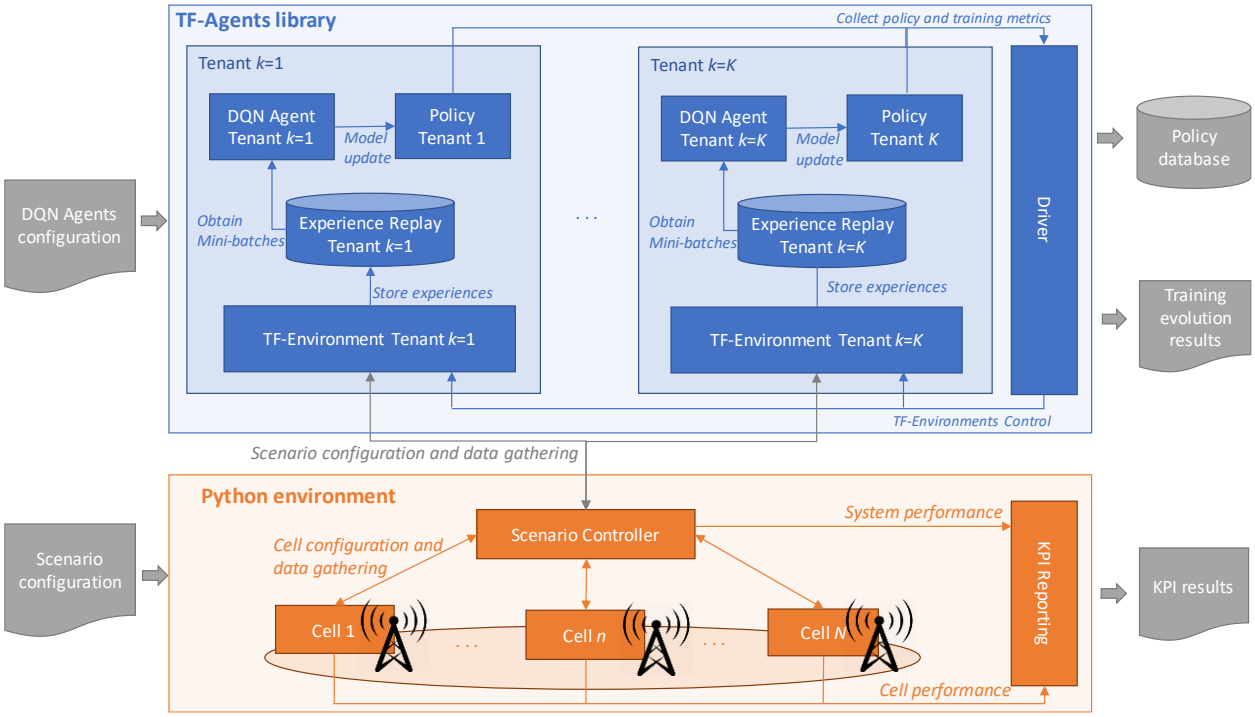


Fig. 63. DRL-based capacity sharing solution development scheme for training.

The solution gets as input parameters the DQN Agents configuration, which includes the DQN Agents hyperparameters, as well as the scenario configuration, which includes the number of cells and tenants included in the solution, the service profile parameters for each of the tenants, the cells' configuration as well as the parameters for the per-cell offered load evolutions. The hyperparameters can be set by performing the tuning, which is held by testing diverse combinations of values of the different DQN Agents hyperparameters and selecting that combination of values that have the best behaviour according to some criteria (e.g., lowest number of steps to obtain a loss variation lower than a threshold as in Section 5.2). During the execution of the developed solution, the learnt policies can be obtained as well as training evolution results (e.g., reward, loss) and the KPI evolution (e.g., SLA satisfaction, throughput, assigned capacity), which are usually obtained periodically to monitor the evolution of training.

Note that the self-developed Python environment has been considered in the solution in order to easily validate and evaluate the DRL-based capacity sharing solution through simulations during the development phase. The developed environment provides a full control and customized configuration of the network,

which has allowed studying the behaviour of the DRL-based capacity sharing solution under a wide range of scenarios and network conditions, which is not feasible to be performed on a real network environment.

Fig. 64 illustrates the developed scheme for the evaluation of the already learnt polices stored in the policy database during the training phase. In this case, the policies of each tenant can be loaded and tested in a target evaluation scenario of interest. To this end, a TF-Environment is created for each tenant as an interface between the tenant's policy and the Python environment, since actions and states cannot be triggered directly from the policy to the Python environment. The driver is also included to manage the different TF-Environments. The Python environment for evaluation is the same as the one during training and it is configured according to the scenario and offered loads conditions desired for evaluation. In this case, only KPI results from the scenario are obtained as an output. Notice that the scheme for evaluation simulates the inference phase of the solution, but instead of applying the learnt policies in the real network, they are applied in a controlled simulated network environment.

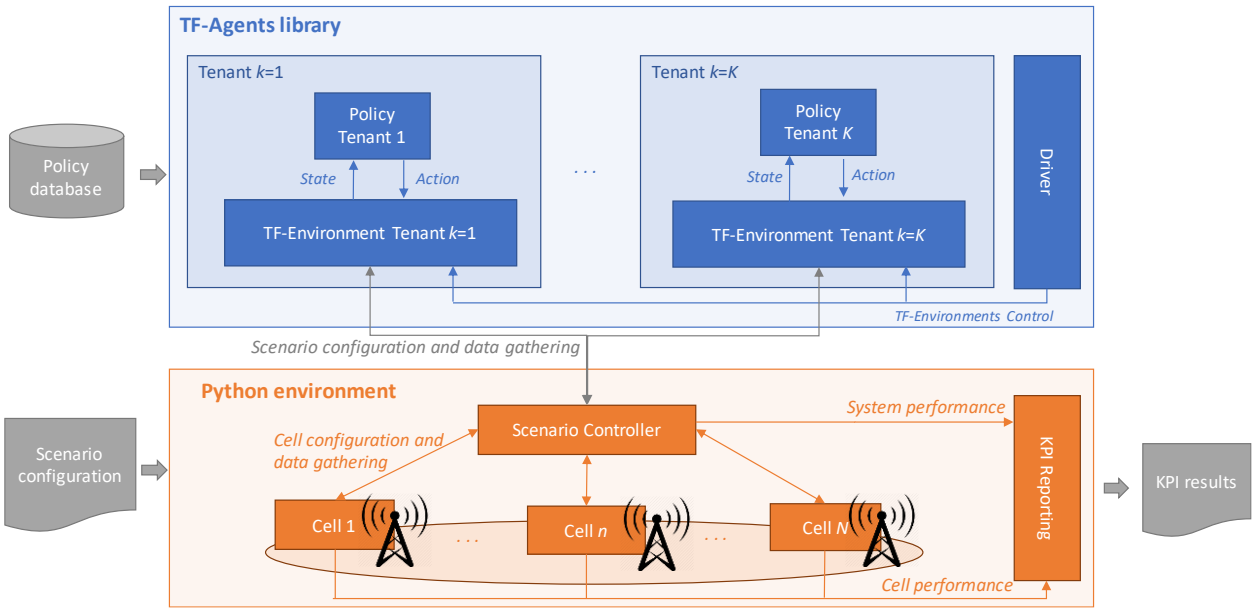


Fig. 64. DRL-based capacity sharing solution development scheme for evaluation.

5.4.2. Production phase of the DRL-based capacity sharing solution for RAN slicing scenarios

The deployment of DRL-based solutions in real production environments requires their preparation to be run in a target production framework and be provided with the required interfaces to interact with the real environment. In the case of the DRL-based capacity sharing solution in Chapter 4, the compatibility requirements with the 3GPP and O-RAN systems discussed in Section 5.2 need to be considered. On the one hand, the developed DRL-based capacity sharing solution requires to be executed during the inference of the learnt policies on the ML inference host of the SMO platform of the O-RAN architecture. To this end and considering the current virtualization trends for Virtual Network Functions (VNFs), the DRL-based capacity sharing solution is containerized by using Dockers. On the other hand, the O1 interface for the communication to configure the O-DU NF of gNBs and obtain performance measurements needs to be implemented according to the O-RAN and 3GPP specifications. In the following, the description of

containers and the details of the O1 interface that need to be developed are provided. Finally, the section ends with the description of the DRL-based capacity sharing solution adopted for production using containers and the required connections integrated in the O1 interface.

5.4.2.1. Containerization

Building upon the ETSI NFV reference architecture [105], the O-RAN architecture considers the cloudification of network functions, where O-RAN network functions and its components can be deployed as Virtual Network Functions (VNFs) in the form of Virtual Machines (VMs) or containers on commercial off-the shelf (COTS) hardware [106]. Only the virtualization of the O-RU function remains as a future study item. Although VMs are extensively used, containers are gaining attention for the deployment of VNFs. In fact, current developments within the O-RAN Software Community [107], which work in the development of O-RAN functions, rely on containers.

Containerization is a virtualization approach through which applications are run as isolated software units called containers, all using the same shared operation system (OS) and managed by a container engine. Fig. 65 shows the high-level architecture of a containerized solution. Each container packages up all the required code, libraries, binaries, configuration files and dependencies required by the application to be run. This is encapsulated and isolated so that the application can be easily deployed and is portable from one computing environment to another [108]. Containers are lightweight since they do not contain OS, which allows their quickly deployment.

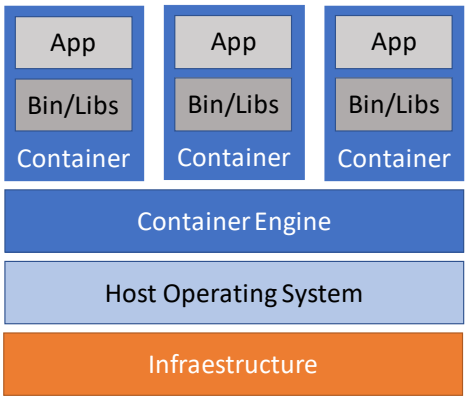


Fig. 65. High-level architecture of a containerized solution.

The container engine is responsible for the automatization of the applications deployment inside software containers. In this regard, Docker is a popular open-source container engine based on Linux containers, which has been widely adopted for the deployment of containers. The architecture of Docker consists in a client-server scheme that allows building container images, which contain code and dependencies of the application, and running them as containers in a standardized manner. In addition, the option to distribute the created containers is available [109]. The Docker client and server communicate according to a provided Docker REST API, and the deployment of applications consisting of multiple containers can be performed through Docker-compose.

Based on the above, the lightweight, agile, and scalable features of containers make them a potential technology for VNF deployment since they allow more easily adapting to the rapidly evolving infrastructure of virtualized RAN [110]. Hence, containers are identified as a potential technology for the production solution of the DRL-based capacity sharing, since they would ease the deployment and operation of the solution in a production environment, e.g., a neutral host where different SMO applications are running. Fig. 66 illustrates the particularization of a container for the DRL-based capacity sharing solution. The app in this case corresponds to the DRL-based capacity sharing code. To be able to run the code in the container, the container performs the installation of Python and all the libraries required to run the code, such as TF-Agents. Note that in the case of using Docker as a container engine, the Docker container for the DRL-based capacity sharing would be executed by firstly creating a container image for the DRL-based capacity sharing container. This image would be generated by creating a "Dockerfile" specifying the version of Python and the code to execute in the container, jointly with the libraries and the respective versions to install. Then, the DRL-based capacity sharing container would be built based on this image.

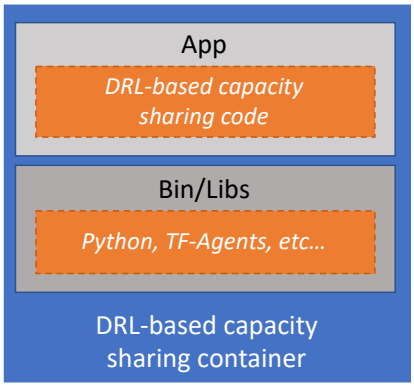


Fig. 66. Illustrative container components for the DRL-based capacity sharing solution.

5.4.2.2. O1 Interface implementation

Within the O-RAN architecture depicted in Fig. 53, the O1 interface is between the SMO and the O-RAN Network Functions (RIC, O-CU-CP, O-CU-UP, O-DU, O-RU) for providing Fault, Configuration, Accounting, Performance and Security (FCAPS) support. The standardization of the O1 interface in [111] identifies diverse services that can be provided through the interface. For the different services, two main roles are distinguished: Management Service (MnS) Consumer and the MnS Producer. Regarding the DRL-based capacity sharing solution in Fig. 56, two main services are identified be able to configure the RRMPolicyRatio attribute per NSSI of the O-DU of each cell and gather the performance measurements from the O-DU in each cell through O1 interface. Note that, for both services the MnS Consumer corresponds to the SMO and the MnS consumer to the O-DU in each cell.

The first service corresponds to the *Provisioning Management Service*, which allows that a MnS Consumer configures the attributes of the managed objects on the MnS Producer. To create, delete, modify and read Managed Object Instances (MOI) on the MnS Provider, the NETCONF protocol is used and changes in the MOI are notified to MnS Consumer through REST/HTTP protocol. The NETCONF protocol defines a

simple mechanism through which a network device can be managed, configuration data information can be retrieved, and new configuration data can be uploaded and manipulated. It has a simple layered architecture as shown in Fig. 67, where the core is a simple Remote Procedure Call (RPC) layer transported over secure transports such as SSH, TLS, SOAP or BEEP, being SSH mandatory [112]. The protocol operates according to a service-client scheme, where the MnS Provider has the role of the server and the MnS Consumer the role of the client. Moreover, NETCONF defines a set of base protocol operations, which include the edit-config operation that allows the MnS Consumer to modify the configuration of MnS Producer parameters through a XML-encoded file that specifies the desired values of the configuration parameters. The configurations of the MnS Producer are defined by using YANG, which is a data modelling language for NETCONF that allow expressing the structure and semantics of configuration information in a vendor-neutral format and in a readable and compact way. In YANG, the definitions are structured in modules, which are provided with a name and a XML namespace, and the configuration parameters are defined in leafs or lists. YANG comes with YIN, which is a XML representation of YANG so that standard XML tools can be used to process YANG data models definitions.

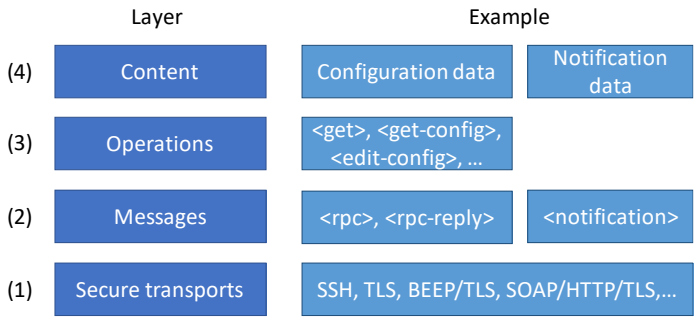


Fig. 67. NETCONF protocol layers.

By using the edit-config operation of NETCONF, MnS Consumer can modify the MOI attributes of MOI in the MnS Producer according to the procedure depicted in Fig. 68. After establishing the NETCONF session, the MnS Consumer sends a RPC request to edit the configuration of the MOI attributes at the MnS Provider according to the YANG module definitions of the MOI. Then, the MnS Provider replies whether confirming the MOI configuration or notifying an error. Finally, the NETCONF connection terminates. Regarding the DRL-based capacity sharing solution, the SMO (MnS Consumer) could modify the *rRMPolicyDedicatedRatio* attribute for each NSSI in the O-DU at each cell (MnS Producer) based on this procedure.

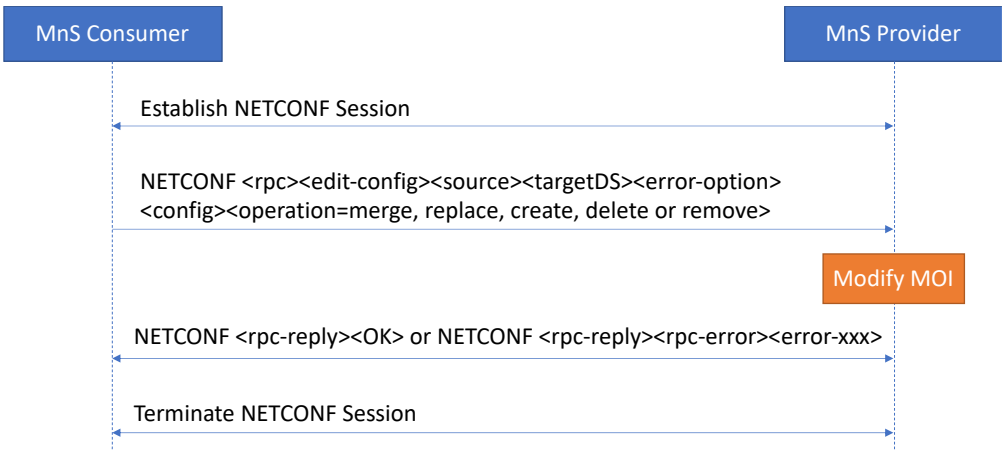


Fig. 68. Simplified Modify MOI Attributes procedure based on [111].

The second service to be implemented for the O1 interface for the DRL-based capacity sharing solution deployment is the *Performance Assurance Management Services*, which allows a MnS Provider to report Performance Measurements (PM) data to a MnS consumer. Performance data can be provided through PM data files or PM Data streaming modes through the procedures in Fig. 69 and Fig. 70, respective. In the case of PM data file reporting, the MnS Provider sends an asynchronous notifyFileReadyNotification to the MnS Consumer over HTTP/TLS when a new PM data file is available. This notification includes the name and location of the new PM data file. The PM data file consists in a XML-based file that is defined according to the 3GPP formats defined in [82] and PM definitions in [41]. Then, the MnS Consumer can retrieve the file from the specified location through SFTP or STPeS. In the case of PM Data Streaming, the MnS Provider requests to establish a WebSocket connection to begin streaming PM data and provides MetaData about the streams that will be sent on the connection. Next, the MnS Consumer accepts the request to upgrade the connection to a WebSocket and the MnS Provider starts transmitting binary encoded data to the MnS Consumer while the connection is active.

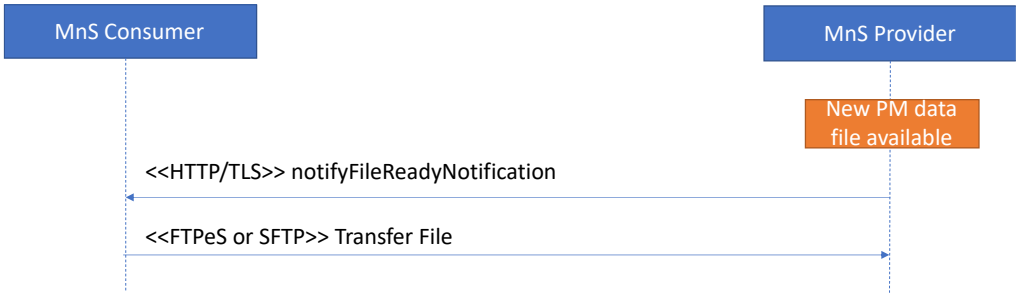


Fig. 69. PM Data File Reporting.

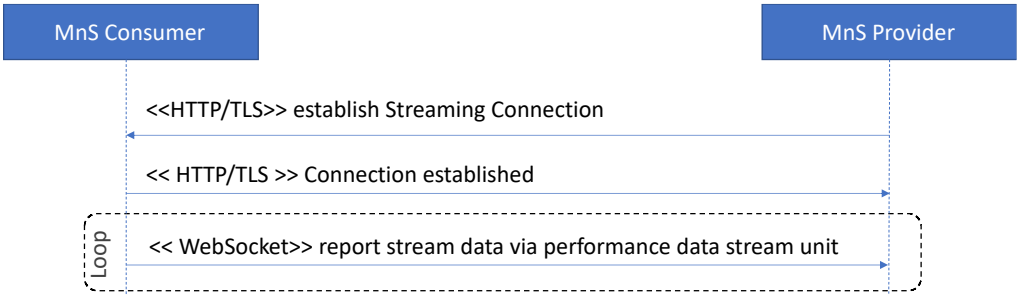


Fig. 70. PM Data Streaming.

5.4.2.3. Production solution description

Based on the above components, the design of the production solution of the DRL-based capacity sharing is presented in this section. Fig. 71 depicts the designed production solution that allows the inference of the learnt policies by the DRL-based capacity sharing solution in a production environment. The code for the DRL-based capacity sharing solution is packed as a Docker container located in the ML inference host, where the container includes all the associated libraries, such as Python or TF-agents, and required dependencies for the operation of the DRL-based capacity sharing code, as illustrated in Fig. 66. This container is responsible for the continuous operation of all those steps required to infer the learnt policies to configure capacity sharing solution in the different cells of the scenario. Moreover, the container has access to a database with the already learnt policies to load them for the inference. The policies in the database correspond to those trained during the development phase as described in Section 5.4.1 but also those resulting from possible re-training of the initial learnt policies if low performance of the initially trained policies is detected.

The DRL-based capacity sharing container is able to interact with the O-DU NF of *N* cells through the O1 interface established for each of them. The cells need to be compliant with the information models of 3GPP and the standardization of the O-DU by the O-RAN to be able to establish the O1 interface [113]. The O1 interface for *n*-th cell considers that the DRL-based capacity sharing container and the ODU-NF of the cell have the roles of the MnS Consumer and MnS Providers, respectively, and that the interface is composed of two connections. First, a NETCONF connection to configure the RRMPolicyRatio attribute per NSSI of the O-DU NF of the cell. To support this connection, the O-DU NF of the cell contains a NETCONF server, which stores the configurations according to the YANG models, and a NETCONF client needs to be included in the DRL-based capacity sharing container for each cell to send modification requests of the *RRMPolicyRatio* attribute per NSSI to the O-DU NF NETCONF server according to the procedure in Fig. 68. Second, a HTTP/TLS and SFTP connection is created for the provisioning of performance measurements from the cell to the DRL-based capacity sharing container. The designed production solution considers that the performance measurements are provided according to PM Data File Reporting mode since measurements are required in the order of minutes, so the PM streaming mode could be more appropriate if lower operation times were required. Therefore, the O-DU NF of each cell also contains a PM server that collects PM files that are transferred to the PM client for that cell located at the DRL-based capacity sharing container according to the procedure in Fig. 69.

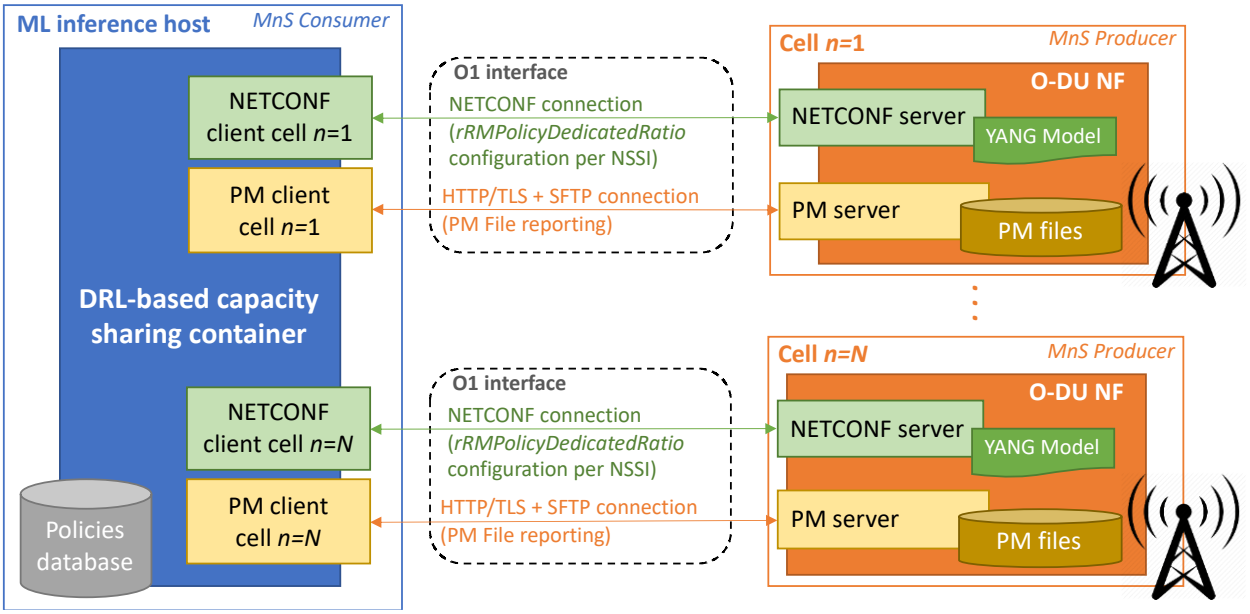


Fig. 71. Containerized solution with O1 interface interaction to *N* cells.

Algorithm 4 summarizes the operations performed by the DRL-based capacity sharing container. Initially, the NETCONF and HTTP/TLS sessions are established between the O-DU NF of each cell n=1…N (lines 1-2 of *Algorithm* 4). As a generality and considering that actions for the NSSI associated to each tenant are obtained independently, a policy for each NSSI is loaded from the policies database (line 3 of *Algorithm* 4). Then, periodically the DRL-based capacity sharing container receives the *notifyFileReady* notification from all cells, which is sent by the PM server of the O-DU NF of each cell when the PM files are available, providing the location from where the PM files can be obtained (line 5 of *Algorithm* 4). The PM files are transferred from the provided location in the *notifyFileReady* to the DRL-based capacity sharing container via SFTP and are processed to compute the state for each agent for all cells (lines 6-7 of *Algorithm* 4). After, the actions for each NSSI are obtained by applying the policy for the computed state (line 8 of *Algorithm* 4). Based on the obtained actions, the *rRMPolicyDedicatedRatio* per NSSI is computed for each cell and is sent through the NETCONF session as a NETCONF modify attribute request through a XML file according to the YANG model in the cell (lines 9-10 of *Algorithm* 4). The operations in lines 5-10 of *Algorithm* 4 are repeated periodically every Δt seconds.

***Algorithm* 4 – DRL-based capacity sharing container operation**	
1	Establish NETCONF session for each cell n=1…N.
2	Establish HTTP/TLS session for each cell n=1…N.
3	Load saved policy from policies database.
4	Periodically (Loop):
5	Receive *notifyFileReady* notification through HTTP/TLS for all cells n=1…N.
6	Obtain PM measurements through SFTP for each cell n=1…N.
7	Compute state from PM measurements for each NSSI.
8	Obtain actions by applying the policies for each NSSI.
9	Obtain the *rRMPolicyDedicatedRatio* based on action for each NSSI and each cell n=1…N.
10	Send *rRMPolicyDedicatedRatio* per NSSI through NETCONF modify attribute request for each cell n=1…N.

Although the scope of this thesis in regard to the production solution of the DRL-based capacity sharing solution is its design and the identification of the main elements and technologies that support it, the first implementation steps of the designed production solution have been taken with the aim to get familiarized with the involved technologies and assess the feasibility and complexity of the designed production solution. Details on the conducted implementation and testing are included in Appendix 2. Moreover, it is worth mentioning that the implementation of the production solution is devised as a future work within the scope of the project "Proof Of concept of a Radio access neTwoRk slicing solution based on Artificial InTelligence (PORTRAIT)" with reference PDC2021-120797-I00 funded by MCIN/AEI/10.13039/501100011033 and by European Union Next GenerationEU/PRTR. This project includes a proof of concept of the DRL-based solution in the 5GCAT pilot [114], which includes the deployment of 5G small cells for testing solutions on a neutral host network and was launched at the end of 2019.

5.5. Summary

Taking as a reference the DRL-based capacity sharing solution presented in the previous chapter, this chapter has discussed on diverse implementation aspects regarding the development and production stages of RL-based solutions for RAN slicing that are essential for its deployment in the real network.

First, Section 5.2 has highlighted the need of defining RL-based solutions for RAN according to the standardization work held by O-RAN and 3GPP. An overview of the main elements of the O-RAN architecture and the application of ML-based assisted solutions in the architecture has been provided, as well as an identification of the 3GPP information models and management services related to capacity sharing in RAN slicing environments. After this, a plausible framework for the integration of the DRL-based capacity sharing solution with the 3GPP information models and management services as well as with the O-RAN architecture and procedures has been proposed, which provides the solution with standardized input and outputs for in real deployments compliant with O-RAN and 3GPP.

Second, in Section 5.3 the relevance of adequately model data used for training RL-based solution has been discussed by conducting an impact analysis of training in the DRL-based capacity sharing solution. To this end, a methodology that allows specifying how to create training datasets with given features to train DRL models for simulation-based training environments has been proposed, jointly with a set of performance metrics that allow assessing the impact of the training datasets on the training and on the inference stages. The methodology has been used to assess the performance the DRL-based capacity sharing solution. Results have shown that: (a) The coverage of states of the training dataset affects the convergence time and the convergence stability: training datasets with low coverages achieve lower convergence times and more stability than datasets with large coverages of states. Specifically, it has been obtained that a dataset with *COV* of about 22% achieves convergence in 75% of the time than a dataset with *COV* of 92%. (b) Training datasets with coverage values of around 25% are sufficient to achieve a satisfactory training process that provides good performing policies. In this respect, it has been able to achieve optimality ratios higher than

0.9 in around 80% of the observed offered load levels during the inference. (c) In general, training datasets with low coverage values below 25% lead to poor performing policies. In this case, the performance is highly dependent on the percentage of coincidence between the offered loads of the training dataset and those experienced during the inference stage.

Third and final, Section 5.4 has focused on the development and production stages of the DRL-based capacity sharing solution. Regarding the development stage, a high-level overview of the developed software solution that relies on the TF-Agents tool both for the training and the evaluation phases has been provided. For the production stage, the required upgrades for the preparation of the developed software solution to be deployed in the real network have been identified. These include the containerization of the solution and the implementation of the standardized interface O1 that allow the interaction of the solution with the network nodes, which relies on the NETCONF protocol for the configuration of network devices and HTTP/TLS and SFTP for the gathering of performance measurements. The design of the DRL-based capacity sharing solution to be deployed in the production environment has been finally described, specifying the operation of the containerized solution with the required interfaces' protocols.

Overall, the implementation aspects treated in this chapter for the DRL-based capacity sharing solution allow providing a practical perspective of the implementation of RL-based solutions for the RAN management in general. This represents relevant contribution of this thesis since solutions for the practical challenges of applying RL-based solutions for the RAN in the real RAN have been discussed and provided.

Chapter 6. Concluding remarks and future directions

6.1. Conclusions

Network slicing has emerged as a key feature of the 5G architecture that will allow multiple tenants to serve their users with the services envisaged for 5G over a common network infrastructure at the same time. The realization of RAN slices is especially challenging since the common pool of radio resources available at the different NG-RAN nodes needs to be adequately managed in order to provide multiple and diverse behaviours associated to the different RAN slices operating on it, so that their requirements are fulfilled and an efficient use of the available resources in the different RAN nodes is achieved. To cope with this complexity, functionalities that allow the proper management of the different slices as well as the configuration the NG-RAN nodes to support RAN slicing are required. Potential solutions to this requirement include the adaptation of existing Radio Resource Management (RRM) functionalities to incorporate RAN slicing parameters and the development of Self-Organizing Network (SON) functions that enable the automatisation of the lifecycle management of RAN slices (i.e., creation, modification and termination of RAN slices), for which Artificial Intelligent (AI) techniques are expected to be a key enabler. In this context, this thesis has contributed to the need of new functionalities for the realization of RAN slicing by defining, developing and evaluating different RRM and AI-based functionalities for RAN slicing. The thesis provides a wide view of the RAN slicing problem, which has allowed identifying the requirements and limitations of RAN slicing solutions at different levels and providing insights into the capabilities of AI-based solutions to deal with RAN slicing from the modelling to the implementation perspective.

Firstly, an assessment of slice-aware RRM functions at different levels of the protocol stack has been performed by means of a Markov model that has allowed the characterisation of multi-tenant and multi-service scenarios (Chapter 2). The model includes RRM functions in terms of admission control at layer 3 and radio resource allocation at layer 2, where both include per-tenant parameters and consider the definition of Guaranteed Bit Rate (GBR) and non-GBR services in terms of their Guaranteed Flow Bit Rate (GFBR), the Allocation and Retention Priority (ARP) indicator and the 5G QoS Identifier (5QI) parameters. The defined radio resource allocation function has been defined by two different approaches: first, in average terms and second, considering a statistical distribution of the spectral efficiency in the cell, which allows capturing the variable propagation conditions of the radio environment. The evaluation of the RRM

function has been performed in a wide range of 5G scenarios of interest (i.e., diverse services requirements, cell deployments, traffic load conditions, etc.) in terms of different performance metrics (blocking probability, degradation probability, throughput, occupation, etc.). Based on the obtained results, the capability of the defined slice-aware RRM functions to achieve isolation in the admission of users and in the allocation of resources among RAN slices has been proven, so that the traffic requirements of a tenant associated to a RAN slice does not affect the achieved performance of the rest of RAN slices. Moreover, the need to consider the radio resource allocation approach that captures the propagation conditions to achieve accurate results in cell scenarios with high variation of the spectral efficiency is found (Appendix 1). Also, the performed studies have shown the capability of the defined slice-aware RRM functions to operate according to the QoS requirements of different services. In this respect, negligible degradation rates are achieved for GBR services, obtaining those GBR services with higher priorities (lower ARP) better performance. Regarding non-GBR services, they are provided with the available resources after the allocation to GBR services according to the priority level parameter in the 5QI. Furthermore, the capability to introduce new tenants in the scenario has been demonstrated, for which the proper re-configuration of the per-tenant parameters is fundamental to achieve high performance of the different tenants operating in the scenario. Considering the above, the usefulness of the proposed Markov model for the evaluation and analytical characterisation of the problem of RAN slicing with regard to RRM functions has been proven, allowing the establishment of relationships between the different dimensions of the RAN slicing problem.

Secondly, this thesis has addressed the problem Slice Admission Control (SAC) by proposing a data analytics-based methodology for the estimation of the required radio resources by a RAN slice (Chapter 3). The proposed methodology can be used by the SAC function to determine the acceptance or rejection of a new RAN slice creation requests. The methodology for estimating radio resources relies on the computation of the probability density function (pdf) of the required radio resources by a RAN slice, which at the same time, relies on the extraction of data analytics information from collected cell performance measurements. The radio resource estimation is carried out in a cell-basis in order to meet the Service Level Agreement (SLA) of both GBR and non-GBR services included in the RAN slice. In addition, the solution has been contextualized within the 3GPP framework for the management of RAN slicing. The evaluation of the proposed methodology has been conducted under various cell deployments, including simulated-based cells and real LTE cell data, and has been applied to different service types, including eMBB services and Mission Critical Push to Talk (MC PTT) services. From the evaluation, evidences of the relevance of using the methodology based on collected measurements from the real cell environment to achieve a proper SAC operation have been provided, achieving more accurate results when the proposed methodology is fed by cell Channel Quality Indicator (CQI) measurements rather than by simulated-based data. Furthermore, evaluations have shown that the methodology can also be used to adapt the RAN slice configuration in emergency scenarios in order to guarantee the SLA of MC PTT services during such situations, which require a high reliability.

Thirdly, this thesis has addressed the problem of capacity sharing in RAN slicing scenarios by proposing a Deep Reinforcement Learning (DRL)-based capacity sharing solution for multi-tenant and multi-cell scenarios (Chapter 4). The DRL-based capacity sharing solution has been approached as a Multi-Agent Reinforcement Learning (MARL) solution, where each agent is associated to a different tenant and learns the policy to tune the capacity assigned to the associated tenant in each of the cells of the scenario based on the Deep Q-Network (DQN) algorithm. Moreover, the solution has been contextualized within the 3GPP management framework as a SON function. The DRL-based capacity sharing solution has been evaluated in a wide range of scenarios with different number of cells and under diverse traffic demands and traffic demands distributions among cells. The potential of the DRL-based capacity sharing solution to adapt the assigned capacity to each tenant to the traffic demands of each of them while achieving high SLA satisfaction and efficient use of the resources available in the different cells has been proved for the considered scenarios for evaluation. Moreover, the solution has achieved a good trade-off between benchmarking capacity sharing schemes, achieving the high SLA satisfaction of capacity sharing solutions with hard-slicing (i.e., a fixed maximum capacity is reserved to each tenant) and the high resource utilization efficiency of non-slicing solutions (i.e., all the capacity is shared among all tenants without reserving any capacity to each tenant). Also, the DRL-based capacity sharing solution has shown a close to optimal performance. Furthermore, the scalability of the solution to easily introduce new tenants has been highlighted by showing the generalization capability of the learnt policy for a tenant to be applied to any other tenant, which allows the introduction of new tenants simply by associating a previously learnt policy to the new tenant.

Finally, implementation considerations for the deployment the DRL-based capacity sharing solution in real network environment have been discussed from different perspectives (Chapter 5), which has allowed identifying the different requirements, challenges and required stages for the deployment of this type of solutions.

One perspective has been the compliance of the DRL-based solution with the standardization work of 3GPP and O-RAN. In this regard, a plausible 3GPP and O-RAN compliant framework that integrates the designed DRL-based capacity solution has been proposed, placing the DRL-based solution within the elements of O-RAN architecture and mapping the parameters of the DRL-based solution with the 3GPP information models parameters related to capacity sharing and 3GPP performance measurements.

Furthermore, implementation aspects have been considered from the perspective of the training by conducting an impact analysis of the data used during the training on the achieved performance of the DRL-based capacity sharing solution during the training and inference (i.e., application of the policies in the environment once trained) stages. To this end, a methodology for creating training datasets with given features has been introduced jointly with a set of performance metrics to conduct the impact analysis. The performed impact analysis for the DRL-based capacity sharing solution has shown that datasets with low coverages (i.e., percentage of included states over the total number of possible states) achieve higher

convergence during training but their performance during the inference have a higher dependence on the similarity between the data included in the training dataset and the experienced during inference. However, training datasets with coverages around 25% were sufficient to achieve high performance during inference. From this impact analysis, the importance of adequately model the data included in training dataset to guarantee the performance of the learnt policies during the inference has been emphasized.

As a final implementation perspective, the development and production stages for the DRL-based capacity sharing solution have been described. For the development stage, a high-level overview of the developed software solution that relies on the tool TF-Agents has been provided. For the production stage, the designed production solution to deploy the solution in a real production environment has been described, which consists in a containerized solution that is able to communicate via the O1 interfaces of the O-RAN architecture, which relies on NETCONF, HTTP and SFTP protocols.

6.2. Future directions

This thesis has designed a production solution for the deployment of the DRL-based capacity sharing solution in Chapter 4 in a real network environment. As future work, the implementation of the production solution is envisaged, which is included in the project "Proof Of concept of a Radio access neTwoRk slicing solution based on Artificial InTelligence (PORTRAIT)" with reference PDC2021-120797-I00 funded by MCIN/AEI/10.13039/501100011033 and by European Union Next GenerationEU/PRTR. In this project, the designed production solution for the DRL-based capacity sharing solution is expected to be tested in the 5GCAT pilot [114]. This will enable the assessment of the solution in a real environment, which will allow assessing its feasibility and identifying further requirements and limitations of the solution from a practical perspective.

Moreover, given that the DRL-based capacity sharing solution has been initially designed to support eMBB services, another future direction is to extend the solution to include other types of services, such as URLLC and mMTC services. These services types typically have strong requirements in terms of latency and reliability. Therefore, adjustments in the definition of the solution, such as the state or reward signals as well as the SLA requirements, should be included to enable the solution to properly behave when services types different to eMBB are considered.

Furthermore, while the current DRL-based capacity sharing solution has been approached by considering MARL, where each agent operates according to DQN, the exploration of the benefits of expanding the current solution to adopt other machine learning techniques is also considered as future work. For instance, upgrades of the DQN algorithm could be considered, such as DDQN or prioritized DQN, which would optimize the training stage but would not contribute to an improvement of the performance of the learnt policies. Also, other machine learning solutions such as Transfer Learning or Federated Learning, which are currently gaining momentum, could be also studied.

Finally, this thesis has identified requirements and limitations for the application of DRL techniques focusing on the problem of capacity sharing in RAN slicing scenarios, providing guidelines for the development and production stages of this kind of solutions. As a future work, the extension of the findings in this thesis with regard to DRL techniques to be applied to other RAN management problems is also devised, which will be of high relevance for the future 6G networks since they are envisaged to be AI-embedded systems.

Appendix 1.Validation of the average spectral efficiency assumption

The results of the Markov model when considering the spectral efficiency S_{eff} as average spectral efficiency that users would observe in the environment for the resource allocation have been compared with the output of a system-level simulator that allows defining realistic scenarios where different environments can be configured in terms of cell deployment, propagation conditions and mobility. User's sessions are generated following a Poisson distribution, while session durations are modelled by an exponential distribution. The AC and resource allocation procedures implemented in the simulator follow the same principles as in the Markov model in Chapter 2, thus placing the focus on assessing the impact that propagation, velocity and traffic load have in terms of predicted performance metrics. The simulator operates on a discrete-time basis with time steps of 1 s. Simulation statistics are measured by averaging discrete samples taken during the simulation time.

TABLE XXVII. MODEL CONFIGURATION PARAMETERS.

Parameter	**Value**	
Number of available PRBs (N_{ava})	51 PRB	
PRB Bandwidth (B)	360 kHz	
Spectral Efficiency (S_{eff})	8.5 b/s/Hz	
Data rate per PRB	3 Mb/s/PRB	
Cell Maximum number of users	$U_{max,s,n}$=50 users for s,n=1,2	
Cell total capacity	156 Mb/s	
Scenario 1: Standard		
Average session generation rate	Tenant 1	varied from 0.001 to 0.12 sessions/s (corresponds to a variation of 0.6Mb/s to 82.8 Mb/s)
	Tenant 2	0.05 session/s (corresponds to 6.2 Mb/s)
Average session duration	120 s	
Maximum capacity threshold	Tenant 1	$C_{max,1}$=93.6Mb/s (corresponds to the 60% of the total capacity)
	Tenant 2	$C_{max,2}$=31.2 Mb/s (corresponds to the 20% of the total capacity)
Tenant generation distribution	Tenant 1	40% of session arrivals are of service 1 and 60% of service 2
	Tenant 2	20% of session arrivals are of service 1 and 80% of service 2
Scenario 2: Disaster		
Average session generation rate	Tenant 1	0.04 session/s (corresponds to 17.76 Mb/s)
	Tenant 2	varied from 0.001 to 0.25 sessions/s (corresponds to a variation of 0.36 Mb/s to 88 Mb/s)
Average session duration	120 s	
Maximum capacity threshold	Tenant 1	$C_{max,1}$=46.8 Mb/s (corresponds to the 30% of the total capacity)
	Tenant 2	$C_{max,2}$=78 Mb/s (corresponds to the 50% of the total capacity)
Tenant generation distribution	Tenant 1	10% of session arrivals are of service 1 and 90% of service 2
	Tenant 2	60% of session arrivals are of service 1 and 40% of service 2

TABLE XXVIII. ENVIRONMENT'S CELL CONFIGURATION.

Environment	UMi	UMa	RMa
ISD (Inter-Site distance)	200 m	500 m	1735 m
gNB height	10 m	25 m	35 m
UE height	1.5 m	1.5 m	1.5 m
Minimum gNB-UE distance	10 m	35 m	35 m
Path Loss and Shadowing model	Model of sec. 7.4 of [45].		
Shadowing standard deviation in Line of Sight (LOS)	4	4	4
Shadowing standard deviation in Non-Line of Sight (NLOS)	7.82	6	8
Frequency	3.6 GHz		
Total gNB transmitted power	44 dBm	49 dBm	52 dBm
gNB antenna Gain	Omnidirectional antenna with 5 dBi gain		
UE noise figure	9 dB		
Link-level model to map Signal to Interference and Noise Ratio (SINR) and bit rate	Model in section A.F of [48] with maximum spectral efficiency of 9.96 b/s/Hz (corresponding to SINR=30 dB) and minimum SINR=-10 dB		
Average Spectral Efficiency	8.5 b/s/Hz	6.9 b/s/Hz	6.3 b/s/Hz
Cell total capacity	156 Mb/s	126.7 Mb/s	115.6 Mb/s
Maximum Capacity Threshold Tenant 1 ($C_{max,1}$)	93.6 Mb/s	76 Mb/s	69.4 Mb/s
Maximum Capacity Threshold Tenant 2 ($C_{max,2}$)	31.2 Mb/s	25.3 Mb/s	23.1 Mb/s
Simulation duration λ_1 =0.05	$15 \cdot 10^6$ s	$2.5 \cdot 10^6$ s	$8 \cdot 10^5$ s
Simulation duration λ_1 =0.1	$7.5 \cdot 10^6$ s	$1.25 \cdot 10^6$ s	$4 \cdot 10^5$ s

To study the validity of the Markov model, the simulator has been configured according to the parameters of the scenario 1 described in Table XXVII. Three different environments are considered: Urban Micro-cell (UMi), Urban Macro-cell (UMa) and Rural Macro-cell (RMa) [45]. The configuration of each environment is detailed in Table XXVIII. Additionally, pedestrian (3 km/h), urban (30 km/h) and high-speed (120 km/h) mobility patterns have been studied, where the User Equipment (UE) position is updated every 5 s following a random-walk model. The simulation duration has been set to ensure the observation of at least 100 blocking events for each service along a simulation (except for the MC PTT service that does not experience blockings in any of the considered cases). In Table XXIX, the results obtained through both the simulator and the Markov model (values in parenthesis) are compared.

In terms of blocking probability, very few differences are found between the Markov model and the simulator for all the studied environments. Indeed, the main assumption affecting the accuracy of the Markov model is the consideration of S_{eff} as the average spectral efficiency that users would observe in the environment. Correspondingly, the comparison in terms of PRB utilisation shows more accurate results for those environments with lower variations on the spectral efficiency (i.e., lower cell range such as, e.g., UMi). Similarly, for the aggregated throughput per slice, the biggest discrepancies are found for the RMa environment where users can experience high spectral efficiency variability.

TABLE XXIX. COMPARISON OF RESULTS OBTAINED VIA THE SYSTEM-LEVEL SIMULATOR IN DIFFERENT SCENARIOS AND THE MARKOV MODEL (IN PARENTHESES).

Env.	Speed (km/h)	λ Tenant 1	PRB utilisation(%)				Blocking probability (%)				Throughput per slice (Mb/s)	
			eMBB-Premium video	eMBB-Basic Video	MC-Video	MC-PTT	eMBB-Premium video	eMBB-Basic Video	MC-Video	MC-PTT	eMBB	MC
UMi	3	0.05	16.56 (15.36)	7.45 (6.92)	4.13 (3.84)	0.16 (0.15)	0.06 (0.06)	0.21 (0.21)	0.12 (0.13)	0 (0)	34.96 (34.78)	6.25 (6.23)
		0.1	32.02 (29.79)	13.19 (12.23)	4.17 (3.85)	0.16 (0.15)	3.12 (3.15)	11.67 (11.68)	0.15 (0.13)	0 (0)	65.83 (65.58)	6.3 (6.25)
	30	0.05	15.76 (15.36)	7.09 (6.92)	3.93 (3.84)	0.15 (0.15)	0.06 (0.06)	0.21 (0.21)	0.12 (0.13)	0 (0)	34.96 (34.78)	6.25 (6.23)
		0.1	30.46 (29.79)	12.56 (12.23)	3.96 (3.85)	0.15 (0.15)	3.12 (3.15)	11.67 (11.68)	0.15 (0.13)	0 (0)	65.83 (65.58)	6.3 (6.25)
UMa	3	0.05	21.49 (18.79)	9.63 (8.42)	5.47 (4.7)	0.21 (0.18)	0.96 (0.83)	1.36 (1.37)	0.98 (0.86)	0 (0)	34.58 (34.48)	6.3 (6.18)
		0.1	38.71 (33.79)	15.27 (13.21)	5.31 (4.7)	0.21 (0.18)	10.57 (10.81)	22.52 (22.58)	0.85 (0.87)	0 (0)	59.99 (59.54)	6.12 (6.19)
	30	0.05	16.56 (15.36)	7.45 (6.92)	4.13 (3.84)	0.16 (0.15)	0.06 (0.06)	0.21 (0.21)	0.12 (0.13)	0 (0)	34.96 (34.78)	6.25 (6.23)
		0.1	32.02 (29.79)	13.19 (12.23)	4.17 (3.85)	0.16 (0.15)	3.12 (3.15)	11.67 (11.68)	0.15 (0.13)	0 (0)	65.83 (65.58)	6.3 (6.25)
RMa	120	0.05	15.76 (15.36)	7.09 (6.92)	3.93 (3.84)	0.15 (0.15)	0.06 (0.06)	0.21 (0.21)	0.12 (0.13)	0 (0)	34.96 (34.78)	6.25 (6.23)
		0.1	30.46 (29.79)	12.56 (12.23)	3.96 (3.85)	0.15 (0.15)	3.12 (3.15)	11.67 (11.68)	0.15 (0.13)	0 (0)	65.83 (65.58)	6.3 (6.25)

From the mobility point of view, under a given environment higher speeds present a better match in the throughput and PRB utilisation results obtained with the simulator and the Markov model. This is because in environments with higher speed, more variated samples in terms of user position are obtained, which contributes to reach more averaged results, presenting more similarities to the ones obtained through the Markov model.

Bearing the above results in mind, it is concluded that the theoretical model provides accurate performance results in a good number of representative realistic environments. For environments with larger fluctuation on S_{eff}, the model requires to be further extended by considering S_{eff} as a random variable with a certain statistical distribution.

Appendix 2. Initial implementation and testing for the production stage

This appendix contains the first implementation steps of the designed production solution for the DRL-based capacity sharing solution, which aim at familiarizing with the technologies of containers, specifically by using Docker, and the interaction of the O1 interface of the O-RAN that rely on NETCONF and SFTP protocols.

The implementation presented here corresponds to a basic communication through the O1 interface using Docker containers. The DRL-based capacity sharing container in Fig. 71 requires to include a NETCONF and SFTP clients for each of the cells in the environment in order to be able to configure the *rRMPolicyDedicatedRatio* per NSSI and obtain performance measurements. In this regard, two Python scripts that play the role of NETCONF and SFTP clients have been created, which use the Python libraries *ncclient* and *pysft*, respectively. At the cell side, given that a real cell is not available for testing, a simple O-DU NF simulator with the minimum functionality of the O-DU NF in a real cell has been implemented for the testing of the O1 interaction, even though the implementation of the O-DU NF is out of the scope of the production solution of the DRL-based capacity sharing solution.

The O-DU NF simulator contains a NETCONF server container, which has been built based on the image of an NETCONF server image available in [115] as a part of the ONAP NF simulator. The considered NETCONF server container allows loading YANG modules and performing operations for reading configurations (get-config operation) and editing them via XML files (edit-config operation). The considered NETCONF server container relies on two more containers called “kafka”, which is used for the messaging with the NETCONF server and the subscription to YANG modules, and “zookeeper”, for the storage of configurations of the YANG module. Moreover, the O-DU NF simulator contains a SFTP server to store PM measurement files and make them available. To easily visualize and upload PM files in the SFTP server, a Filezilla container is used, which is a SFTP client with a user interface. In order to build the different containers in the O-DU NF simulator in a practical manner, a Docker-compose file has been created that indicates, for each container, the image to build, the ports to interact with the container, the volumes of the container in the local server and the dependencies between containers. By executing the Docker-compose file, all the containers in the O-DU NF can be started at the same time (i.e., the containers do not need to be started one by one). Considering the above, Fig. 72 shows the list of the different containers that compose the O-DU NF simulator after running the Docker-compose file, which correspond

to those grouped under the stack called “docker_poc”. Note that an additional container called “portainer” is also running, which is the platform that is used to visualize and manage the different containers.

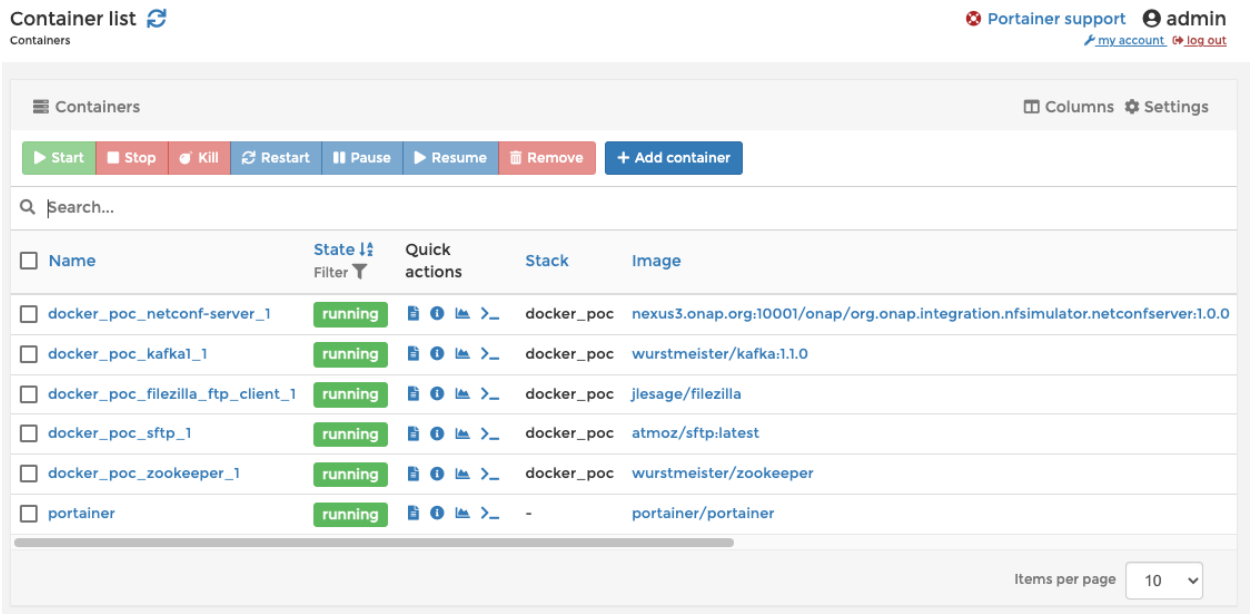


Fig. 72. Screenshot of the running containers that compose the O-DU NF simulator.

In order to test the NETCONF connection, the NETCONF server has been provided with the standardised YANG module for the RRMpolicyratio object (see in Section 5.2.2) called “_3gpp-nr-nrm-rrmpolicy.yang”, which is available in the section E.5.26 of 3GPP TS 28.541. This YANG module defines the attributes (i.e., elements to configure) that can be configured for the RRMPolicyRatio object. For the case of the DRL-based capacity sharing solution, the attributes to configure correspond to the *rRMpolicyDedicatedRatio*, which is defined for a *rRMPolicyMemberList*. The *rRMPolicyMemberList* contains the *RRMPolicyMember*(*s*), defined by the *sNSSAI* information (i.e., the Slice/Service type (SST) and the Slice Differentiator (SD) as defined in 3GPP TS 23.003) and the *pLMNId* information (i.e., the Mobile Country Code (MCC) and the Mobile Network Code (MNC)).

To configure the YANG model, the sample XML file in Fig. 73 has been created, which configures the *rRMpolicyDedicatedRatio* of two tenants, denoted as Tenant 1 and Tenant 2, by defining two *RRMPolicyRatio* elements, each for one tenant. The *rRMpolicyDedicatedRatio* is established to 60% and 40% to Tenant 1 and Tenant 2, respectively. Note that the *rRMpolicyMaxRatio* and *rRMpolicyMinRatio* are defined although they are not needed since they must be defined, which are set to 100% and 0%, respectively, for both tenants. This configuration has been loaded to the NETCONF server container as initial configuration of the yang module _3gpp-nr-nrm-rrmpolicy.yang. Also, it is worth mentioning that for the creation of the XML file, a sample XML file based on the module _3gpp-nr-nrm-rrmpolicy.yang has been created by using the tool *pyang* and the correctness of XML file for the _3gpp-nr-nrm-rrmpolicy.yang has been validated by using a tool called *yang-tools*.

To test the NETCONF communication with the NETCONF server, a simple NETCONF client that is able to obtain the current configuration of the module _3gpp-nr-nrm-rrmpolicy.yang and modify it has been developed in Python. This NETCONF client relies on the Python library *ncclient* that allows sending requests to a NETCONF server, such as the get-config and edit-config operations tested here.

```
<?xml version='1.0' encoding='UTF-8'?>
<RRMPolicyRatio xmlns="urn:3gpp:sa5:_3gpp-nr-nrm-rrmpolicy">
  <id>1</id>
  <attributes>
    <resourceType>PRB</resourceType>
    <rRMPolicyMemberList>
      <mcc>214</mcc>
      <mnc>01</mnc>
      <sd>FFFFFF</sd>
      <sst>1</sst>
    </rRMPolicyMemberList>
    <rRMPolicyMaxRatio>100</rRMPolicyMaxRatio>
    <rRMPolicyMinRatio>0</rRMPolicyMinRatio>
    <rRMPolicyDedicatedRatio>60</rRMPolicyDedicatedRatio>
  </attributes>
</RRMPolicyRatio>
<RRMPolicyRatio xmlns="urn:3gpp:sa5:_3gpp-nr-nrm-rrmpolicy">
  <id>2</id>
  <attributes>
    <resourceType>PRB</resourceType>
    <rRMPolicyMemberList>
      <mcc>214</mcc>
      <mnc>02</mnc>
      <sd>FFFFFF</sd>
      <sst>1</sst>
    </rRMPolicyMemberList>
    <rRMPolicyMaxRatio>100</rRMPolicyMaxRatio>
    <rRMPolicyMinRatio>0</rRMPolicyMinRatio>
    <rRMPolicyDedicatedRatio>40</rRMPolicyDedicatedRatio>
  </attributes>
</RRMPolicyRatio>
```

Fig. 73. Sample XML file for the configuration of module _3gpp-nr-nrm-rrmpolicy.yang.

Firstly, the get-config operation of the NETCONF client has been tested. Fig. 74 includes the NETCONF client code in Python that has been developed for the get-config operation to obtain the current configuration of the *rRMpolicyDedicatedRatio* of Tenant 1 and Tenant 2 in the NETCONF server. As an output of this script, the XML configuration in Fig. 75 is obtained, where the *rRMpolicyDedicatedRatio* of Tenant 1 and Tenant 2 are 60% and 40%, respectively. This configuration corresponds to the initial configuration in Fig. 73 that had been loaded to the NETCONF server as the initial configuration, which proves that the NETCONF client is able to get the current configuration of the *rRMpolicyDedicatedRatio* in the NETCONF server satisfactorily. Secondly, the edit-config operation of the NETCONF client has been tested. Fig. 76 includes the developed NETCONF client code for the edit-config operation to modify the *rRMpolicyDedicatedRatio* of Tenant 1 and Tenant 2 to 10% both in the NETCONF server. After running this script and, afterwards, executing again the get-config code in Fig. 74, the new configuration in Fig. 77 has been obtained, which shows that current configuration of the *rRMpolicyDedicatedRatio* for Tenant 1

and Tenant 2 to 10% both as configured in the edit-config code. This indicates that the edit-config operation has been performed satisfactorily.

```
from ncclient import manager

Host_IP='XX.XX.XX.XX'
Port='830'
User='XXXX'

subtree_filter_RRMPolicy_dedicated_ratio_all='''<RRMPolicyRatio
xmlns="urn:3gpp:sa5:_3gpp-nr-nrm-
rrmpolicy"><id/><attributes><rRMPolicyDedicatedRatio/></attributes></RRMPolicyRatio>'''

def demo(host, port, user, password,expr):
    with manager.connect(host=host, port=port, username=user, password=password,
hostkey_verify=False) as m:
        assert(":xpath" in m.server_capabilities)
        c = m.get_config(source='running', filter=('subtree',expr)).data_xml
        with open("%s_test_get_config.xml" % host, 'w') as f:
            f.write(c)

if __name__ == '__main__':
    demo(Host_IP, Port,User, User, subtree_filter_RRMPolicy_dedicated_ratio_all)
```

Fig. 74. NETCONF client Python code for get-config operation to obtain the *rRMPolicyDedicatedRatio.*

```
<?xml version="1.0" encoding="UTF-8"?>
- <data xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  - <RRMPolicyRatio xmlns="urn:3gpp:sa5:_3gpp-nr-nrm-rrmpolicy">
      <id>1</id>
    - <attributes>
        <rRMPolicyDedicatedRatio>60</rRMPolicyDedicatedRatio>
      </attributes>
    </RRMPolicyRatio>
  - <RRMPolicyRatio xmlns="urn:3gpp:sa5:_3gpp-nr-nrm-rrmpolicy">
      <id>2</id>
    - <attributes>
        <rRMPolicyDedicatedRatio>40</rRMPolicyDedicatedRatio>
      </attributes>
    </RRMPolicyRatio>
</data>
```

Fig. 75. Obtained configuration after the execution of the code in Fig. 74.

```
from ncclient import manager

Host_IP='XX.XX.XX.XX'
Port='830'
User='XXXX'
Pw='****'

val1=10
val2=10


def demo(host, port, user, password, v1, v2):
  snippet ='''<config xmlns="urn:ietf:params:xml:ns:netconf:base:1.0"><RRMPolicyRatio
xmlns="urn:3gpp:sa5:_3gpp-nr-nrm-
rrmpolicy"><id>1</id><attributes><rRMPolicyDedicatedRatio>%d</rRMPolicyDedicatedRatio></a
ttributes></RRMPolicyRatio><RRMPolicyRatio xmlns="urn:3gpp:sa5:_3gpp-nr-nrm-
rrmpolicy"><id>2</id><attributes><rRMPolicyDedicatedRatio>%d</rRMPolicyDedicatedRatio></a
ttributes></RRMPolicyRatio></config>''' % (v1,v2)

  with manager.connect(host=host, port=port, username=user, password=password,
hostkey_verify=False) as m:
    assert(":validate" in m.server_capabilities)
    m.edit_config(target='running', config=snippet, format = 'xml',test_option="set")
```

Fig. 76. NETCONF client Python code for edit-config operation to modify the *rRMPolicyDedicatedRatio* configuration.

```
<?xml version="1.0" encoding="UTF-8"?>
- <data xmlns:nc="urn:ietf:params:xml:ns:netconf:base:1.0" xmlns="urn:ietf:params:xml:ns:netconf:base:1.0">
  - <RRMPolicyRatio xmlns="urn:3gpp:sa5:_3gpp-nr-nrm-rrmpolicy">
      <id>1</id>
    - <attributes>
        <rRMPolicyDedicatedRatio>10</rRMPolicyDedicatedRatio>
      </attributes>
    </RRMPolicyRatio>
  - <RRMPolicyRatio xmlns="urn:3gpp:sa5:_3gpp-nr-nrm-rrmpolicy">
      <id>2</id>
    - <attributes>
        <rRMPolicyDedicatedRatio>10</rRMPolicyDedicatedRatio>
      </attributes>
    </RRMPolicyRatio>
</data>
```

Fig. 77. Obtained configuration by executing Fig. 74 running the edit-config code in Fig. 76.

Regarding the SFTP connection, a test has been conducted that consists in a SFTP client that downloads and processes PM measurements from the SFTP server. In the SFTP server, a PM file that includes the required measurements to run the DRL-based capacity sharing solution is generated. These performance measurements are based in those standardized in 3GPP specification TS 28.552 and are the following:

- "Mean DL PRB used for data traffic": This measurement provides the number of physical resource blocks (PRBs) in average used in downlink for data traffic, which can be provided per supported S-NSSAI. The measurement is needed for the computation of the occupation of a tenant in a cell included in the state definition of the DRL-based capacity sharing solution. The name of the measurement in the PM files is RRU.PrbUsedDl.SNSSAI.

- "DL total available PRB": This measurement provides the total number of physical resource blocks (PRBs) in average available downlink. The measurement is needed for the computation of the PRBs available in the cell not used by any tenant included in the state definition of the DRL-based capacity sharing solution. The name of the measurement in the PM files is RRU.PrbAvailDl.

- "DL PDCP PDU Data Volume": This measurement provides the Data Volume (amount of PDCP PDU bits) in the downlink delivered from GNB-CU to GNB-DU, which can be provided per supported S-NSSAI. The measurement is required for the computation of the throughput of a tenant in a the cell, which is used by the solution to compute the capacity sharing solution. The name of the measurement in the PM file is QosFlow.PdcpPduVolumeDL_Filter.

The PM file that includes these measurements is formatted according to the XML schema file "measData.xsd" available in the 3GPP specification TS 28.550, which contains the validation tools and the format of the PM XML files. Moreover, the TS 28.550 specifies the different fields in the "measData.xsd". Considering this format, a XML PM file has been created manually that is shown in Fig. 78. In the PM file, the PM of two tenants, Tenant 1 and Tenant 2, are provided where the performance measurements referring to each of them are indicated by a ".1" and ".2", respectively (i.e. the RRU.PrbUsedDl for Tenant 1 is RRU.PrbUsedDl.1), as indicated to the specification in TS 28.552. Therefore, the values in the PM file are RRU.PrbUsedDl.1=20 PRBs, RRU.PrbUsedDl.2=25 PRBs, RRU.PrbAvailDl= 20 PRBS, QosFlow.PdcpPduVolumeDL.1= 32327 Mbits and QosFlow.PdcpPduVolumeDL.2= 40751 Mbits.

Moreover, the PM file indicates that the measurements started on the 1/12/2021 at 8:00 and finished at 1/12/2021 at 8:15, with a duration of 900 seconds, as indicated by the "PT900S" code in the PM file.

```
<?xml version="1.0" encoding="utf-8"?>
<md:MeasDataFile xmlns:md="http://www.3gpp.org/ftp/specs/
archive/28_series/28.550#measData" xmlns:xsi="http://www.w3.org/2001/XMLSchema" xsi
:schemaLocation="http://www.3gpp.org/ftp/specs/archive/28_series/28.550#measData
measData.xsd">
  <md:fileHeader fileFormatVersion="28.550 V16.0" vendorName="" dnPrefix="">
    <md:fileSender senderName="Cell1" senderType="O-DU" />
    <md:MeasData beginTime="2021-12-01T08:00:00" />
  </md:fileHeader>
  <md:measData>
    <md:measuredEntity userLabel="NRCell0" localDn="" swVersion="" />
    <md:measInfo measInfoId="RRU_block">
      <md:job jobId="job0001" />
      <md:granPeriod duration="PT900S" endTime="2021-12-01T08:15:00" />
      <md:repPeriod duration="PT900S" />
      <md:measType p="1" >RRU.PrbUsedDl.1</md:measType>
      <md:measType p="2" >RRU.PrbUsedDl.2</md:measType>
      <md:measType p="3" >RRU.RRU.PrbAvailDl</md:measType>
      <md:measValue measObjLdn="">
       <md:r p="1">20</md:r>
       <md:r p="2">25</md:r>
       <md:r p="3">20</md:r>
        <md:suspect>true</md:suspect>
      </md:measValue>
    </md:measInfo>
    <md:measInfo measInfoId="QoSFlow_block">
      <md:job jobId="job0002" />
      <md:granPeriod duration="PT900S" endTime="2021-12-01T08:15:00" />
      <md:repPeriod duration="PT900S" />
      <md:measType p="4" >QosFlow.PdcpPduVolumeDL.1</md:measType>
      <md:measType p="5" >QosFlow.PdcpPduVolumeDL.2</md:measType>
      <md:measValue measObjLdn="">
       <md:r p="4">32327</md:r>
       <md:r p="5">40751</md:r>
        <md:suspect>true</md:suspect>
      </md:measValue>
    </md:measInfo>
  </md:measData>
  <md:fileFooter>
    <md:MeasData endTime="2021-12-01T08:15:00" />
  </md:fileFooter>
</md:MeasDataFile>
```

Fig. 78. PM file generated according to the format in measData.xsd in 3GPP TS 28.550.

To perform the testing of the SFTP connection, this XML PM file has been uploaded to the SFTP server container under the file name *xml_PM_report_simple.xml* and a simple SFTP client that downloads and processes the file has been developed in Python by using the libraries *pysftp*, which supports the SFTP connection, and *xml*, which allows processing the XML file. Fig. 79 includes an screenshot of the Filezilla container that shows that at the remote site folder (in the right), which corresponds to the SFTP server container, the *xml_PM_report_simple.xml* is available to download. The developed code for the SFTP client is shown in Fig. 80, which firstly establishes the SFTP connection with the SFTP server, then downloads the *xml_PM_report_simple.xml* file and finally reads the file and outputs the configuration of the different performance metrics of interest. These outputs are displayed in Fig. 81, which proves that the SFTP client is able to download the PM file and read the required parameters names and values. While this test displays these values in the screen, the values could be used for the inference of the DRL-based capacity sharing solution.

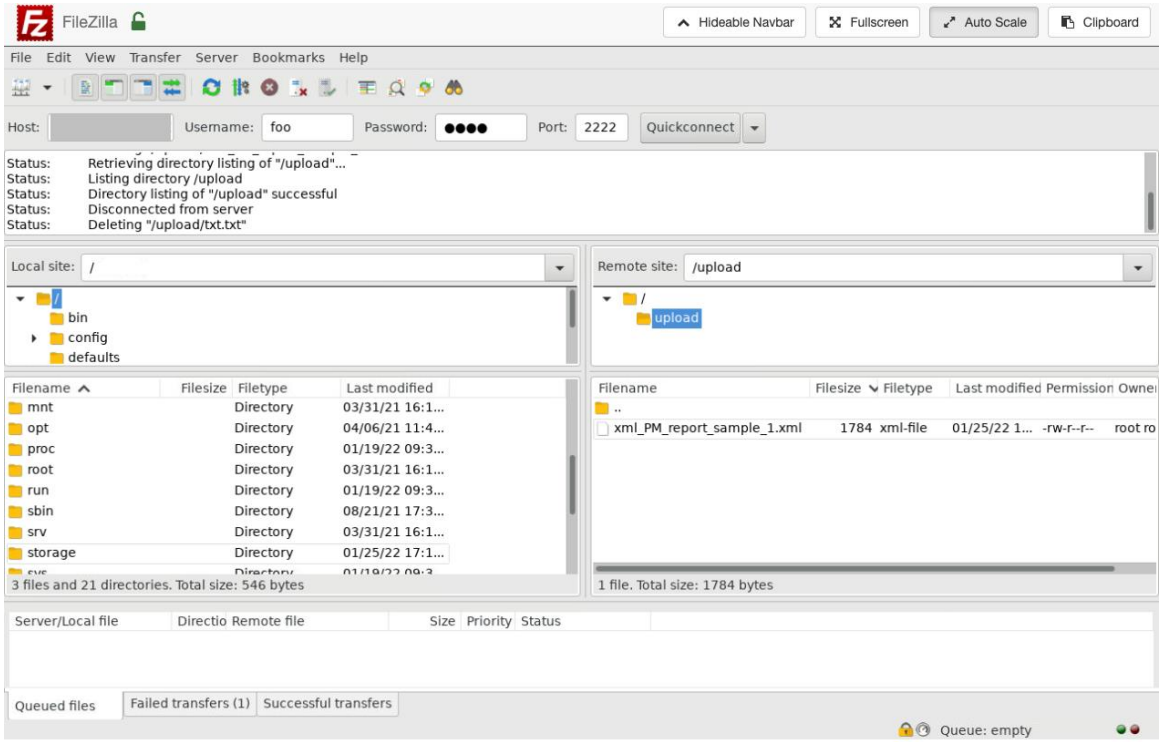


Fig. 79. Screenshot of the Filezilla SFTP client showing that the SFTP server has the PM file.

```python
import pysftp, os
import xml.etree.ElementTree as ET

#ESTABLISH CONNECTION TO SFTP SERVER
with pysftp.Connection(host=Hostname, username=Username, password=Password, port=Port) as sftp:
    print("Connection successfully established ... ")
    # Switch to a remote directory
    sftp.cwd('/upload/')

    directory_structure = sftp.listdir_attr()

    remoteFilePath = directory_structure[-1].filename #Read last file
    # Use get method to download a file
    sftp.get(remoteFilePath)

file_name='xml_PM_report_sample_1.xml'

# Passing the path of the xml document to enable the parsing process
tree = ET.parse(file_name)

# getting the parent tag of the xml document
root = tree.getroot()

#Register namespace
ET.register_namespace('md',"{http://www.3gpp.org/ftp/specs/archive/28_series/28.550#measData}")

#Namespace from xml
ns={'md' : "http://www.3gpp.org/ftp/specs/archive/28_series/28.550#measData"}

#Print type and value:
print(root.find(".//md:measType[@p='1']",ns).text,":",root.find(".//md:r[@p='1']",ns).text)
print(root.find(".//md:measType[@p='2']",ns).text,":",root.find(".//md:r[@p='2']",ns).text)
print(root.find(".//md:measType[@p='3']",ns).text,":",root.find(".//md:r[@p='3']",ns).text)
print(root.find(".//md:measType[@p='4']",ns).text,":",root.find(".//md:r[@p='4']",ns).text)
print(root.find(".//md:measType[@p='5']",ns).text,":",root.find(".//md:r[@p='5']",ns).text)
```

Fig. 80. SFTP client code for downloading the XML file and process it.

```
RRU.PrbUsedDl.1 : 20
RRU.PrbUsedDl.2 : 25
RRU.RRU.PrbAvailDl : 20
QosFlow.PdcpPduVolumeDL.1 : 32327
QosFlow.PdcpPduVolumeDL.2 : 40751
```

Fig. 81. Output to the code in Fig. 80.

To sum up, the performed implementations have allowed testing the NETCONF and SFTP connections as well as identifying and gaining knowledge of the different tools and protocols that are involved in both connections. The development of the O-DU NF simulator, which includes the containers for the NETCONF and SFTP servers, has involved working with Dockers, for which has been found particularly useful the use of Docker-compose file to start running the different containers together, as well as stopping them to make modifications. Moreover, insights into the NETCONF protocol and usage of YANG models previously defined by 3GPP and the generation of XML files that configure them have been provided. In this regard, the use of *pyang* and *yang-tools* has been fundamental for the creation and validation of XML files for configuration of the YANG module. Also, the performed testing has included the definition of PM files, defined by XML schemas in .xsd format by 3GPP. The use of both the configuration YANG modules and the generation of PM files have implied strong efforts to analyse the 3GPP standards. Finally, to create the NETCONF and SFTP client codes, the use of the existing *ncclient*, *pysftp* and *xml* libraries has notably eased the development due to the large availability of documentation and examples, making them a feasible option for the development of designed production solution of the DRL-based capacity sharing solution, included as a future work in the scope of the PORTRAIT project (reference PDC2021-120797-I00).

References

[1] NGMN Alliance, “5G White Paper,” February 2015.

[2] ITU-R, “IMT Vision-Framework and overall objectives of the future development of IMT for 2020 and beyond”. Recommendation ITU-R M.2083, September 2015.

[3] K. Samdanis, X. Costa-Perez and V. Sciancalepore, "From network sharing to multi-tenancy: The 5G network slice broker," in *IEEE Communications Magazine*, vol. 54, no. 7, pp. 32-39, July 2016.

[4] 3GPP TS 38.300 V15.3.1, “NR; NR and NG-RAN overall description; Stage 2 (Release 15),” October 2018.

[5] 3GPP TS 23.501 V15.3.0, “System architecture for the 5G system; Stage 2 (Release 15),” September 2018.

[6] E. Dahlman, S. Parkvall, J. Sköld, *5G NR; The next generation wireless access technology*, Academic Press, Elsevier, 2018.

[7] J. Ordonez-Lucena, P. Ameigeiras, D. Lopez, J. J. Ramos-Munoz, J. Lorca and J. Folgueira, "Network Slicing for 5G with SDN/NFV: Concepts, Architectures, and Challenges," in *IEEE Communications Magazine*, vol. 55, no. 5, pp. 80-87, May 2017.

[8] P. Rost, et al. “Mobile network architecture evolution toward 5G,” in *IEEE Communication Magazine*, May, 2016.

[9] 5GPP Architecture Working Group, “View on 5G Architecture,” December 2017.

[10] I. Afolabi, T. Taleb, K. Samdanis, A. Ksentini and H. Flinck, "Network Slicing and Softwarization: A Survey on Principles, Enabling Technologies, and Solutions," in *IEEE Communications Surveys & Tutorials*, vol. 20, no. 3, pp. 2429-2453, thirdquarter 2018, doi: 10.1109/COMST.2018.2815638.

[11] 3GPP TR 28.801 V15.1.0, “Study on management and orchestration of network slicing for next generation network” (Release 14), April 2018.

[12] R. Ferrus, O. Sallent, J. Perez-Romero and R. Agusti, "On 5G Radio Access Network Slicing: Radio Interface Protocol Features and Configuration," in *IEEE Communications Magazine*, vol. 56, no. 5, pp. 184-192, May 2018.

[13] P. Rost et al., "Network slicing to enable scalability and flexibility in 5G mobile networks," in *IEEE Communications Magazine*, vol. 55, no. 5, pp. 72-79, May 2017.

[14] S. E. Elayoubi, S. B. Jemaa, Z. Altman and A. Galindo-Serrano, "5G RAN Slicing for Verticals: Enablers and Challenges," in *IEEE Communications Magazine*, vol. 57, no. 1, pp. 28-34, January 2019.

[15] O. Sallent, J. Perez-Romero, R. Ferrús, R. Agusti, "On radio access network slicing from a radio resource management perspective," in *IEEE Wireless Communications*, October, 2017, pp. 166-174.

[16] 3GPP TS 28.531 v16.2.0, “Management and orchestration; Provisioning; (Release 16),” June 2019.

[17] 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*, vol. 20, no. 5, pp. 1333-1346, Oct. 2012.

[18] X. Costa-Perez, J. Swetina, T. Guo, R. Mahindra and S. Rangarajan, "Radio access network virtualization for future mobile carrier networks," in *IEEE Communications Magazine*, vol. 51, no. 7, pp. 27-35, July 2013.

[19] A. Ksentini and N. Nikaein, "Toward enforcing network slicing on RAN: flexibility and resources abstraction," in *IEEE Communications Magazine*, vol. 55, no. 6, pp. 102-108, June 2017.

[20] 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*, vol. 25, no. 5, pp. 3044-3058, October 2017.

[21] B. Khodapanah, A. Awada, I. Viering, D. Oehmann, M. Simsek and G. P. Fettweis, "Fulfillment of Service Level Agreements via Slice-Aware Radio Resource Management in 5G Networks," *2018 IEEE 87th Vehicular Technology Conference (VTC Spring)*, Porto, June, 2018, pp. 1-6.

[22] E. Pateromichelakis and C. Peng, "Selection and Dimensioning of Slice-Based RAN Controller for Adaptive Radio Resource Management," *2017 IEEE Wireless Communications and Networking Conference (WCNC)*, San Francisco, CA, May 2017, pp. 1-6.

[23] E. Pateromichelakis and K. Samdanis, ‘‘A graph coloring based inter-slice resource management for 5G dynamic TDD RANs,’’ in *Proc. IEEE Int. Conf. Commun. (ICC)*, Kansas City, MO, USA, May 2018, pp. 1–6.

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

[25] D. Bega, M. Gramaglia, A. Banchs, V. Sciancalepore and X. Costa-Perez, "A Machine Learning approach to 5G Infrastructure Market optimization," in *IEEE Transactions on Mobile Computing*, Feb., 2019.

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

[27] Foukas, Xenofon, Mahesh K. Marina, and Kimon Kontovasilis. "Orion: RAN slicing for a flexible and cost-effective multi-service mobile network architecture." in *23rd Annual International Conference on Mobile Computing and Networking*, *ACM*, 2017.

[28] J. Epperlein and J. Mareček, "Resource allocation with population dynamics," in *2017 55th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, Monticello, IL, October 2017, pp. 1293-1300.

[29] S. Jagannatha, N. S. Shravan and S. Kavya, "Cost performance analysis: Usage of resources in cloud using Markov-chain model," in *2017 4th International Conference on Advanced Computing and Communication Systems (ICACCS)*, Coimbatore, January 2017, pp. 1-8.

[30] H. Y. Ng, K. T. Ko and K. F. Tsang, "3G mobile network call admission control scheme using Markov chain," in *Proceedings of the Ninth International Symposium on Consumer Electronics, June 2005. (ISCE 2005)*., 2005, pp. 276-280.

[31] X. Gelabert, J. Pérez-Romero, O. Sallent and R. Agustí, "A Markovian approach to radio access technology selection in heterogeneous multiaccess/multiservice wireless networks," in *IEEE Transactions on Mobile Computing*, vol. 7, no. 10, pp. 1257-1270, Oct. 2008.

[32] V. V. Paranthaman, Y. Kirsal, G. Mapp, P. Shah and H. X. Nguyen, "Exploring a new proactive algorithm for resource management and its application to wireless mobile environments," in *2017 IEEE 42nd Conference on Local Computer Networks (LCN)*, Singapore, Oct.2017, pp. 539-542.

[33] S. Al-Rubaye, A. Al-Dulaimi, J. Cosmas and A. Anpalagan, "Call admission control for non-standalone 5G ultra-dense networks," in *IEEE Communications Letters*, vol. 22, no. 5, pp. 1058-1061, May 2018.

[34] Y. Kim and S. Park, "Analytical Calculation of Spectrum Requirements for LTE-A Using the Probability Distribution on the Scheduled Resource Blocks," in *IEEE Communications Letters*, vol. 22, no. 3, pp. 602-605, March 2018.

[35] B. Han, D. Feng and H. D. Schotten, "A Markov Model of Slice Admission Control," in *IEEE Networking Letters*, vol.1, no. 1, pp. 2-5, March 2019.

[36] M. N. Patwary, R. Abozariba and M. Asaduzzaman, "Multi-operator spectrum sharing models under different cooperation schemes for next generation cellular networks," *2017 IEEE 86th Vehicular Technology Conference (VTC-Fall)*, Toronto, ON, Sept 2017, pp. 1-7.

[37] S. Lin et al., "Advanced dynamic channel access strategy in spectrum sharing 5G systems," in *IEEE Wireless Communications*, vol. 24, no. 5, pp. 74-80, Oct. 2017.

[38] K. B. Ali, M. S. Obaidat, F. Zarai and L. Kamoun, "Markov model-based adaptive CAC scheme for 3GPP LTE femtocell networks," *2015 IEEE International Conference on Communications (ICC)*, London, June 2015, pp. 6924-6928.

[39] A. Jee, S. Hoque, B. Talukdar and W. Arif, "Analysis of Link Maintenance Probability for Cognitive Radio Ad Hoc Networks," *2018 5th International Conference on Signal Processing and Integrated Networks (SPIN)*, Noida, 2018, pp. 385-389.

[40] J. Pérez-Romero, O. Sallent, R. Ferrús and R. Agustí, "Self-optimised admission control for multitenant radio access networks," *2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, Montreal, QC, Oct. 2017, pp. 1-5.

[41] 3GPP TS 28.552 v16.2.0, “Management and orchestration; 5G Performance measurements (Release 16),” June 2019.

[42] 3GPP TS 28.533 v15.1.0, “Management and orchestration; Architecture framework (Release 16),” June 2019.

[43] 3GPP TS 38.214 v15.6.0, “ NR; Physical layer procedures for data (Release 15),” June 2019.

[44] W.J. Stewart, *Introduction to the Numerical Solution of Markov Chains*, Princeton: Princeton University Press, 1994.

[45] 3GPP TR 38.901 v15.5.0, “Study on channel model for frequencies from 0.5 to 100GHz (Release 15),” March 2019.

[46] 3GPP TS 38.104 v15.5.0, “NR; Base Station (BS) radio transmission and reception (Release 15),” April 2019.

[47] 3GPP TS 38.211 v15.3.0, “NR; Physical channels and modulation (Release 15),” Sept. 2018.

[48] 3GPP TR 38.803 v14.2.0, “Study on new radio access technology: Radio Frequency (RF) and co-existence aspects (Release 14),” Sept. 2017.

[49] J. Ordonez-Lucena et al., "The Creation Phase in Network Slicing: From a Service Order to an Operative Network Slice," 2018 European Conference on Networks and Communications (EuCNC), Ljubljana, Slovenia, 2018, pp. 1-36.

[50] B. Han, D. Feng, L. Ji, and H. D. Schotten, “A Profit-Maximizing Strategy of Network Resource Management for 5G Tenant Slices,” arXiv:1709.09229, 2017.

[51] R. Challa, V. V. Zalyubovskiy, S. M. Raza, H. Choo and A. De, "Network Slice Admission Model: Tradeoff Between Monetization and Rejections," in *IEEE Systems Journal*, vol. 14, no. 1, pp. 657-660, March 2020.

[52] 3GPP TS 28.541 v16.0.0, “Management and orchestration; 5G Network Resource Model (NRM) (Release 16),” March 2019.

[53] A. S. D. Alfoudi, S. H. S. Newaz, A. Otebolaku, G. M. Lee and R. Pereira, "An Efficient Resource Management Mechanism for Network Slicing in a LTE Network," in *IEEE Access*, vol. 7, pp. 89441-89457, 2019

[54] D. Marabissi and R. Fantacci, "Highly Flexible RAN Slicing Approach to Manage Isolation, Priority, Efficiency," in *IEEE Access*, vol. 7, pp. 97130-97142, 2019.

[55] P. L. Vo, M. N. H. Nguyen, T. A. Le and N. H. Tran, "Slicing the Edge: Resource Allocation for RAN Network Slicing," in *IEEE Wireless Communications Letters*, vol. 7, no. 6, pp. 970-973, Dec. 2018.

[56] J. Pérez-Romero, O. Sallent, R. Ferrús and R. Agustí, "Profit-Based Radio Access Network Slicing for Multi-tenant 5G Networks," 2019 *European Conference on Networks and Communications (EuCNC)*, Valencia, Spain, 2019, pp. 603-608.

[57] Ö. U. Akgül, I. Malanchini and A. Capone, "Dynamic Resource Trading in Sliced Mobile Networks," in *IEEE Transactions on Network and Service Management*, vol. 16, no. 1, pp. 220-233, March 2019.

[58] J. Shi, H. Tian, S. Fan, P. Zhao and K. Zhao, "Hierarchical Auction and Dynamic Programming Based Resource Allocation (HA&DP-RA) Algorithm for 5G RAN Slicing," 2018 *24th Asia-Pacific Conference on Communications (APCC)*, Ningbo, China, 2018, pp. 207-212.

[59] J. Gang and V. Friderikos, "Optimal resource sharing in multi-tenant 5G networks," 2018 *IEEE Wireless Communications and Networking Conference (WCNC)*, Barcelona, 2018, pp. 1-6.

[60] 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*, vol. 27, no. 2, pp. 662-675, April 2019.

[61] J. Pérez-Romero, O. Sallent, R. Ferrús and R. Agustí, "Self-optimized admission control for multitenant radio access networks," 2017 *IEEE* 28th *Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, Montreal, QC, 2017, pp. 1-5.

[62] Z. Xiong, Y. Zhang, D. Niyato, R. Deng, P. Wang and L. Wang, "Deep Reinforcement Learning for Mobile 5G and Beyond: Fundamentals, Applications, and Challenges," in *IEEE Vehicular Technology Magazine*, vol. 14, no. 2, pp. 44-52, June 2019.

[63] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, et al.,

"Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[64] H. van Hasselt, et. al, "Deep Reinforcement Learning with Double Q-Learning," *Thirtith AAAI Conference on Artificial Intelligence (AAAI-16)*, USA, 2016.

[65] T. Shaul, et. al, "Prioritized experience replay," *4th International Conference on Learning Representations (ICLR 2016)*, USA, 2016.

[66] M. Hessel, et. al, "Rainbow: Combining Improvements in Deep Reinforcement Learning," *Thirty-Second AAAI Conference on Artificial Intelligence (AAAI-2018)*, USA, 2018.

[67] T. P. Lillicrap, et. al, "Continuous control with deep reinforcement learning," arXiv:1509.02971, Sep. 2015.

[68] V. Mnih, et. al, "Asynchronous Methods for Deep Reinforcement Learning," *33th International Conference on Machine Learning (ICML 2016)*, USA, 2016.

[69] R. Li et al., "Deep Reinforcement Learning for Resource Management in Network Slicing," in *IEEE Access*, vol. 6, pp. 74429-74441, 2018.

[70] C. Qi, Y. Hua, R. Li, Z. Zhao and H. Zhang, "Deep Reinforcement Learning With Discrete Normalized Advantage Functions for Resource Management in Network Slicing," in *IEEE Communications Letters*, vol. 23, no. 8, pp. 1337-1341, Aug. 2019.

[71] Y. Hua, R. Li, Z. Zhao, X. Chen and H. Zhang, "GAN-Powered Deep Distributional Reinforcement Learning for Resource Management in Network Slicing," in *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 2, pp. 334-349, Feb. 2020.

[72] G. Sun, Z. T. Gebrekidan, G. O. Boateng, D. Ayepah-Mensah and W. Jiang, "Dynamic Reservation and Deep Reinforcement Learning Based Autonomous Resource Slicing for Virtualized Radio Access Networks," in *IEEE Access*, vol. 7, pp. 45758-45772, 2019.

[73] G. Sun, K. Xiong, G. O. Boateng, D. Ayepah-Mensah, G. Liu and W. Jiang, "Autonomous Resource Provisioning and Resource Customization for Mixed Traffics in Virtualized Radio Access Network," in *IEEE Systems Journal*, vol. 13, no. 3, pp. 2454-2465, Sept. 2019.

[74] T. Li, X. Zhu and X. Liu, "An End-to-End Network Slicing Algorithm Based on Deep Q-Learning for 5G Network," in *IEEE Access*, vol. 8, pp. 122229-122240, 2020.

[75] G. Sun, G. O. Boateng, D. Ayepah-Mensah, G. Liu and J. Wei, "Autonomous Resource Slicing for Virtualized Vehicular Networks With D2D Communications Based on Deep Reinforcement Learning," in *IEEE Systems Journal*, vol. 14, no. 4, pp. 4694-4705, Dec. 2020.

[76] J. Mei, X. Wang, K. Zheng, G. Bondreau, A. Bin, H. Abou-zeid, "Intelligent Radio Access Network Slicing for Service Provisioning in 6G: A Hierarchical Deep Reinforcement Learning Approach," in *IEEE Transactions on Communications* (Early Access), June, 2021.

[77] V. García, "Deep reinforcement learning based approaches for capacity sharing in radio access network slicing," Master Thesis, Universitat Politècnica de Catalunya, July, 2020.

[78] Y. Abiko, T. Saito, D. Ikeda, K. Ohta, T. Mizuno and H. Mineno, "Flexible Resource Block Allocation to Multiple Slices for Radio Access Network Slicing Using Deep Reinforcement Learning," in *IEEE Access*, vol. 8, pp. 68183-68198, 2020.

[79] Y. Abiko, T. Saito, D. Ikeda, K. Ohta, T. Mizuno and H. Mineno, "Radio Resource Allocation Method for Network Slicing using Deep Reinforcement Learning," in *2020 International Conference on Information Networking (ICOIN)*, 2020, pp. 420-425.

[80] L. Busoniu, R. Babuska, B. De Schutter, "Multi-agent reinforcement learning: An overview," Chapter 7 in *Innovations in Multi-Agent Systems and Applications – 1* (D. Srinivasan and L.C. Jain, eds.), vol. 310 of *Studies in Computational Intelligence*, Berlin, Germany: Springer, pp. 183–221, 2010.

[81] 3GPP TR 28.861 v1.1.0, "Telecommunication management; Study on the Self-Organizing Networks (SON) for 5G networks (Release 16)," Dec. 2019.

[82] 3GPP TS 28.550 v16.7.0, "Management and orchestration; Performance assurance (Release 16)," Dec. 2020.

[83] K. Arulkumaran, M. P. Deisenroth, M. Brundage and A. A. Bharath, "Deep Reinforcement Learning: A Brief Survey," in *IEEE Signal Processing Magazine*, vol. 34, no. 6, pp. 26-38, Nov. 2017.

[84] S. Guadarrama, et. al (2018). TF-Agents: A library for Reinforcement learning in TensorFlow. Available at: https://github.com/tensorflow/agents.

[85] O-RAN Alliance, "O-RAN Minimum Viable Plan and Acceleration Towards Commercialization," White Paper, June 2021.

[86] O-RAN.WG1.O-RAN-Architecture-Description-v05.00, "O-RAN Architecture Description version 5.00," O-RAN Alliance, Working Group 1, Technical specification, July 2021.

[87] O-RAN.WG2.AIML-v01.03, "O-RAN Working Group 2; AI/ML workflow description and requirements v01.03", O-RAN Alliance, Work Group 2, Technical Report, July 2021.

[88] RP-201612, "Study on enhancement of RAN Slicing for NR", 3GPP TSG RAN Meeting #89e, September, 2020.

[89] SP-190931, "Feasibility Study on Enhancement of Network Slicing Phase 2", 3GPP TSG-SA Meeting #85, September, 2019.

[90] SP-200766, "Study on network slice management enhancement", 3GPP TSG-SA Meeting #89e, September, 2020.

[91] SP-200571, "Feasibility Study on Enhanced Access to and Support of Network Slice", TSG SA Meeting #88e, July, 2020.

[92] RWS-210659, "Summary of RAN Rel-18 Workshop Summary", TSG RAN Rel-18 Workshop, June 2021.

[93] GSMA NG.116 - Generic Network Slice Template Version 2.0 (2019-10-16).

[94] 3GPP TS 22.261 v17.3.0, "Service Requirements for the 5G System; Stage 1 (Release 17)", July, 2020.

[95] 3GPP TS 22.104 v17.3.0, "Service requirements for cyber-physical control applications in vertical domains; Stage 1 (Release 17)", July, 2020.

[96] E. d. S. Nascimento, I. Ahmed, E. Oliveira, M. P. Palheta, I. Steinmacher and T. Conte, "Understanding Development Process of Machine Learning Systems: Challenges and Solutions," 2019 *ACM/IEEE International Symposium on Empirical Software Engineering and Measurement (ESEM)*, 2019, pp. 1-6.

[97] L. E. Lwakatare, I. Crnkovic and J. Bosch, "DevOps for AI – Challenges in Development of AI-enabled Applications," 2020 *International Conference on Software, Telecommunications and Computer Networks*

(SoftCOM), 2020, pp. 1-6.

[98] Matthias Plappert (2016). Keras RL. GitHub repository, Available at: https://github.com/keras-rl/keras-rl.

[99] Dhariwal et. al (2017). OpenAI Baselines. GitHub repository. Available at: https://github.com/openai/baselines.

[100] Kuhnle et al. (2017). a TensorFlow library for applied reinforcement learning. Available at: https://github.com/tensorforce/tensorforce.

[101] Brockman, G. et al. (2016). Openai gym. arXiv preprint arXiv:1606.01540.

[102] "Create MATLAB Reinforcement Learning Environments - MATLAB & Simulink". Retrieved December 14, 2021, from https://www.mathworks.com/help/reinforcement-learning/ug/create-matlab-environments-for-reinforcement-learning.html.

[103] 3GPP TS 28.554 v16.5.0, "Management and Orchestration; 5G end to end Key Performance Indicators (KPI) (Release 16)", July, 2020.

[104] Tensorflow. "Inside TensorFlow: TF-Agents." *Youtube*, 9 April 2020, https://www.youtube.com/watch?v=U7g7-Jzj9qo.

[105] ETSI GS NFV 002 V1.2.1, "Network Functions Virtualisation (NFV); Architectural Framework", Dec. 2014.

[106] O-RAN Alliance, "O-RAN Use Cases and Deployment Scenarios. Towards Open and Smart RAN,", White Paper, February 2020.

[107] O-RAN Software Community, O-RAN Alliance, Accessed 14 December 2021, https://wiki.o-ran-sc.org/display/ORAN.

[108] Docker. "Use containers to Build, Share and Run your applications". Accessed 14 December 2021 from https://www.docker.com/resources/what-container.

[109] Dockers. "Docker overview". Retrieved 14 December 2021 from https://docs.docker.com/get-started/overview/.

[110] A. Gopalasingham, D. G. Herculea, C. S. Chen and L. Roullet, "Virtualization of radio access network by Virtual Machine and Docker: Practice and performance analysis," 2017 *IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, 2017, pp. 680-685.

[111] O-RAN.WG10.O1-Interface.0-v05.00, "O-RAN Operations and Maintenance Interface Specification," O-RAN Alliance, Working Group 10, Technical Specification, August 2020.

[112] J. Schönwälder, M. Björklund and P. Shafer, "Network configuration management using NETCONF and YANG," in *IEEE Communications Magazine*, vol. 48, no. 9, pp. 166-173, Sept. 2010, doi: 10.1109/MCOM.2010.5560601.

[113] O-RAN-WG4.CUS.0-v07.0 "Control, User and Synchronization Plane Specification", O-RAN Alliance, Working Group 4, Technical Specification, July 2021.

[114] 5GCAT. "El impulso decisive para una Sociedad digital". Accessed 22 January 2022 from https://www.pilot5gcat.com/es.

[115] ONAP, "NF Simulator". Accessed 27 January 2022 from https://docs.onap.org/projects/onap-integration/en/latest/simulators/nf_simulator.html.