MACHINE LEARNING-ASSISTED CROSS-SLICE RADIO RESOURCE OPTIMIZATION: IMPLEMENTATION FRAMEWORK AND ALGORITHMIC SOLUTION

Ramon Ferrús, Jordi Pérez-Romero, Oriol Sallent, Irene Vilà, Ramon Agustí
Dept. of Signal Theory and Communications, Universitat Politècnica de Catalunya (UPC), c/ Jordi Girona, 1-3, Barcelona, Spain

NOTE: Corresponding author: Ramon Ferrús (ferrus@tsc.upc.edu)

Abstract *– Network slicing is a central feature in 5G and beyond systems to allow operators to customize their networks for different applications and customers. With network slicing, different logical networks, i.e. network slices, with specific functional and performance requirements can be created over the same physical network. A key challenge associated with the exploitation of the network slicing feature is how to efficiently allocate underlying network resources, especially radio resources, to cope with the spatio-temporal traffic variability while ensuring that network slices can be provisioned and assured within the boundaries of Service Level Agreements / Service Level Specifications (SLAs/SLSs) with customers. In this field, the use of artificial intelligence, and, specifically, Machine Learning (ML) techniques, has arisen as a promising approach to cater for the complexity of resource allocation optimization among network slices. This paper tackles the description of a feasible implementation framework for deploying ML-assisted solutions for cross-slice radio resource optimization that builds upon the work conducted by 3GPP and O-RAN Alliance. On this basis, the paper also describes and evaluates an ML-assisted solution that uses a Multi-Agent Reinforcement Learning (MARL) approach based on the Deep Q-Network (DQN) technique and fits within the presented implementation framework.*

Keywords – 5G, cross-slice resource optimization, deep learning, machine learning, network slicing

1. INTRODUCTION

Network slicing allows operators to customize their networks for different applications and customers [1], [2]. Slices can differ in functionality (e.g. air interface capabilities, mobility tracking features), in performance requirements (e.g. latency, availability, reliability and data rates), or they can serve only specific users (e.g. public safety users, corporate customers, or industrial users). A network slice can provide the functionality of a complete network, including radio access network and core network functions. Support for network slicing has been introduced by the 3rd Generation Partnership Project (3GPP) as part of the first release of the Fifth Generation (5G) system specifications (Release 15), with multiple enhancements still to follow in future releases, as reflected by different study items in progress, such as [3]-[6].

The creation and management of network slices is especially challenging in the Radio Access Network (RAN), where multiple slices can be delivered over the same radio channel and the system shall guarantee that the allocation and distribution of the radio resources within the radio channel is done so that specific requirements per slice can be fulfilled (e.g. guaranteed capacity) while using radio resources efficiently [7]-[14]. Remarkably, the automation of the life-cycle management of network slices in the RAN requires two main functionalities: slice admission control and cross-slice resource optimization.

Slice admission control is needed to decide 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. Under Network as a Service (NaaS) business models such as neutral host services, the slice requirements will be determined by the Service Level Agreement (SLA) / Service Level Specifications (SLS) established between the service provider (e.g. the operator of a RAN infrastructure installed in a venue) and the customer (e.g. a Mobile Network Operator - MNO). The fulfillment of the RAN slice requirements may result in the need to guarantee the availability of a certain amount of radio resources to the new slice, defined in terms of, e.g. number of Resource Blocks (RBs) per cell, percentage of cell capacity, etc. Therefore, the slice admission control shall estimate the amount of radio resources required by the new slice and decide whether this can be enforced given the deployed network capacity and the amount of resources consumed by the already admitted slices.

Once multiple slices have been activated in the RAN, the cross-slice resource optimization shall ensure that the slice requirements are satisfied over time and RAN resources are efficiently utilized. This may imply a dynamic modification of the slice configurations (e.g. specifying the amount of radio resources assigned to each slice at each cell, adjustment of slice-aware scheduling settings, configuration of rate limiters, bandwidth parts, mobility load balancing parameters, access control priorities, etc.) during its lifetime in order to deal with the dynamics of the traffic load of the slice and with the random propagation effects that lead to non-deterministic mapping between radio resource consumption and performance requirements. Cross-slice resource optimization has been identified by 3GPP as a use case in the context of Self-Organizing Network (SON) feasibility studies [15], addressing not only the dynamic allocation of radio resources to slices but also the distribution of other resources such as storage and computing for virtualized implementations.

The decision-making logic for cross-slice resource optimization needs to deal with a lot of uncertainties and random processes associated with the variability in traffic generation, device mobility and radio channel conditions, so it is highly difficult to have an accurate a priori statistical knowledge of the network resource utilization and delivered performance. For this reason, model-free Machine Learning (ML)-based methods, which do not rely on predefined models but are able to learn and/or predict the particular network dynamics as well as to operate under goal-oriented policies, become adequate solutions to the problem [16]. Besides, the complexity of the problem with a huge number of variables and conditions (e.g. particular device capabilities, pending traffic, link channel conditions, resource consumption, etc.) also pushes for the introduction of these sorts of methods. As a result, the system can be in a large number of possible states in which the cross-slice resource allocation needs to determine the optimum capacity sharing among slices. In this case, among the possible ML techniques, deep reinforcement learning (RL) schemes become particularly relevant because they provide faster convergence under large state/action spaces in comparison with classical reinforcement learning.

While there is a significant amount of work addressing the cross-slice resource optimization problem from an algorithmic and performance assessment perspective, less attention has been paid to the practical implementation aspects of these solutions, as it will be further discussed in Section 2. In this respect, departing from 3GPP and O-RAN Alliance specifications, a first contribution of this paper is the delineation of the functional framework and information models to be accounted when targeting a practical realization of ML-assisted cross-slice radio resource optimization solutions for 5G and beyond systems. More specifically, the focus is put here on the identification of the specific functional components enabling the deployment of ML-based solutions for RAN management along with the set of information models that have been defined to represent SLAs, network slice instances' characteristics and slicing-related configuration parameters of 5G base stations. On this basis, a second contribution of this paper is the formulation and assessment of a plausible ML-assisted cross-slice radio resource optimization solution that fits within the delineated implementation framework. The solution makes use of Multi-Agent Reinforcement Learning (MARL) based on the Deep Q-Network (DQN) technique. Illustrative performance results of the proposed solution are provided by means of simulations.

The rest of the paper is organized as follows. Section 2 presents an overview of related works in order to position the paper in relation to the state-of-the-art. Section 3 describes the implementation framework, which is particularized to the proposed ML-assisted cross-slice optimization solution in Section 4 and Section 5 presents some illustrative proof-of-concept results. Finally, our concluding remarks are wrapped up in Section 6.

2. RELATED WORK

Artificial Intelligence (AI) and more specifically ML techniques have been applied in the literature for both slice admission control and cross-slice radio resource allocation. In the area of slice admission control, [17] studied an optimal algorithm using Semi-Markov Decision Processes (SMDP) and then proposed an adaptive algorithm based on Q-learning. Then, other works have considered deep Q-learning [18] along with variants for enhancing the training process, such as deep dueling neural networks [19]. ML tools have also been used for enhancing the slice admission control with traffic prediction, such as in [20], [21], which use Holt-Winters prediction, or [22], which uses a combination of Long Short Term Memory (LSTM) and dense neural networks for predicting the resource usage.

In the field of cross-slice optimization, different approaches exist exploiting several ML tools. Q-learning was used in [23] to design a slicing controller that decides which resource units are allocated to each slice based on requirements at the user level. Q-learning complemented with a genetic algorithm was considered in [24] for scaling down allocated resources to slices for congestion control purposes. In [25] deep deterministic policy gradient (DDPG) is used to allocate resource blocks to different tenants in a cloud RAN environment. In turn, game theory with exponential learning is proposed in [26] to divide the network resources (i.e. bandwidth) among slices using OpenFlow, being a general approach not particularized to the specificities of radio resource allocation. Recently, deep Q learning has become a quite popular tool for allocating radio resources to slices, as reflected by works [27]-[33] that include different variants of this technique and address the problem from different perspectives, such as the joint allocation of computational resources and radio resources to users in [27], the allocation of aggregate capacity per slice to multiple cells in [28], [29], the allocation of resources to slices on a single cell basis in [30], [31], [32], or the allocation of per-cell resources to the different slices jointly considering multiple cells in [33]. Finally, other works have proposed the use of traffic forecasting for cross-slice resource allocation, applying techniques such as LSTM neural networks [34], deep convolutional neural networks [35], Generative Adversarial Networks (GANs) [36], or deep neural networks [37].

This paper introduces several novelties with respect to previous works. First of all, this paper presents a functional framework aligned with current 3GPP and O-RAN specifications for implementing ML-assisted cross-slice radio resource optimization and particularizes it to a specific algorithmic solution coming from our previous work [33]. Instead, the above-mentioned works have put the focus on algorithm development but without going into detail of the mapping on existing architectures from standardization bodies. For example, some works just consider a slicing controller (e.g. [23]) or a network slicing module (e.g. [28], [29]) but without providing details of how this would be mapped on practical architectures. Only in [24] an architectural framework for slice management and orchestration that is aligned with 3GPP is presented, but without providing specific details on the algorithm implementation.

Another important novelty comes from the specification of the SLA terms for a RAN slice to be used by the ML-based solution. This paper takes as a reference the attributes defined in the GSMA Generic Slice Template considered by 3GPP to specify the SLA to be fulfilled for a RAN slice across a geographical area covering multiple cells in terms of, e.g. the total amount of capacity to be provided to each slice. Instead, other approaches such as [28]-[32] just consider the SLA specified in terms of the QoS parameters defined at the user level, but without enforcing any aggregate capacity per slice.

Finally, another difference with respect to previous works comes from the algorithmic solution considered in the proposed framework, which consists of a multi-agent DQN with one agent per slice that learns the capacity to be allocated to each slice in each cell. In contrast to single agent solutions like those of [30], [31], which jointly consider all the tenants when making decisions, the multi-agent approach has advantages such as better scalability as it allows easily adding/removing slices in the scenario simply by adding/removing the corresponding agent. Moreover, while some multi-agent approaches have already been considered in [28], [29][32], the one considered here has the advantage that an agent learns the policy for assigning capacity to be provided to the slice in each cell, in contrast to [32], which considered the capacity in a single cell, or [28], [29], which provided the aggregated capacity over all the cells.

3. ML-ENABLED CROSS-SLICE MANAGEMENT FRAMEWORK

3.1 O-RAN framework for ML-assisted solutions

As part of the specification of new interfaces and functionality for an open and intelligent RAN, the O-RAN Alliance is working on the definition of a framework for the deployment of ML-assisted solutions within the RAN (i.e. solutions that rely on the use of ML models such as supervised learning, reinforcement learning, etc.) [38].

A representation of the overall RAN functional architecture being defined by O-RAN is illustrated in Fig. 1 [39]. This constitutes a disaggregated RAN, compliant with 3GPP specifications, where the radio protocol stack is split and distributed between different RAN nodes. In particular, the O-RAN Radio Unit (O-RU) hosts the RF processing and the lower part of the PHY layer functionality (e.g. i/FFT processing), the O-RAN Distributed Unit (O-DU) is in charge of the High-PHY layer processing (e.g. modulation, channel coding), Medium Access Control (MAC) and Radio Link Control (RLC), the O-RAN Central Unit - Control Plane (O-CU-CP) hosts the upper layers of the control plane radio protocol stack, i.e. Radio Resource Control (RRC) and control plane of Packet Data Convergence Protocol (PDCP), and the O-RAN Central Unit - User Plane (O-CU-UP) handles the upper layers of the user plane protocol stack, i.e. Service Data Adaptation Protocol (SDAP) and user plane of PDCP layers. Then, sitting on top of these RAN nodes handling the distributed radio protocol stack, there is the near-real-time RAN Intelligent Controller (near-RT RIC), which serves as the brain of the RAN by coping with the different Radio Resource Management (RRM) functions needed for overall RAN operation, such as radio connection, mobility, Quality of Service (QoS) and interference management. With respect to the interfaces between these RAN nodes, E1, F1-c and F1-u interfaces are specified by 3GPP while Open fronthaul and E2 are being specified by the O-RAN Alliance.

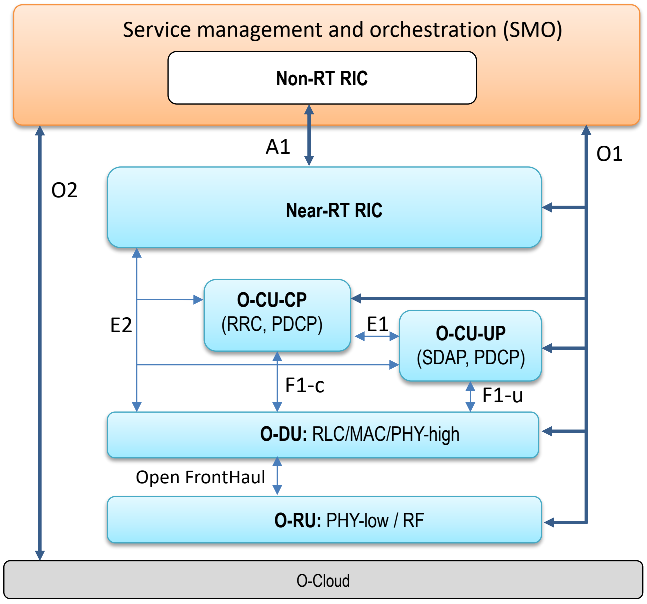

Fig. 1 – O-RAN functional architecture

Moving at the management plane, O-RAN defines the Service Management and Orchestration (SMO) layer, which actually represents the Operations Support Systems (OSS) of the MNO for the RAN domain. As part of the SMO layer, O-RAN basically defines the role of a non-real-time RAN Intelligent Controller (non-RT RIC) entity for the interaction with the near-RT RIC via the A1 interface, which is also being standardized by the O-RAN Alliance. Through the A1 interface [40], the non-RT RIC can perform policy management, ML model management (described below in more detail) and delivery of enriched information for near-RT RIC operation (e.g. RAN data analytics that could be exploited by the near-RT RIC). Furthermore, complementing the A1 interface, the interactions between the SMO and the underlying RAN nodes also rely on the adoption of other standardized interfaces named as O1 and O2 in Fig. 1. In particular, O1 refers to the set of service-based management interfaces being standardized by 3GPP for configuration, performance and fault management of the RAN functionality [41]. In turn, the O2 interface supports the management of the cloud infrastructure and resources allowing the execution of virtualized RAN functions.

Building upon such a RAN reference architecture, Fig. 2 shows the main components and relations being delineated under O-RAN for the training and deployment of ML-assisted solutions within the SMO layer and/or within the RAN nodes themselves.

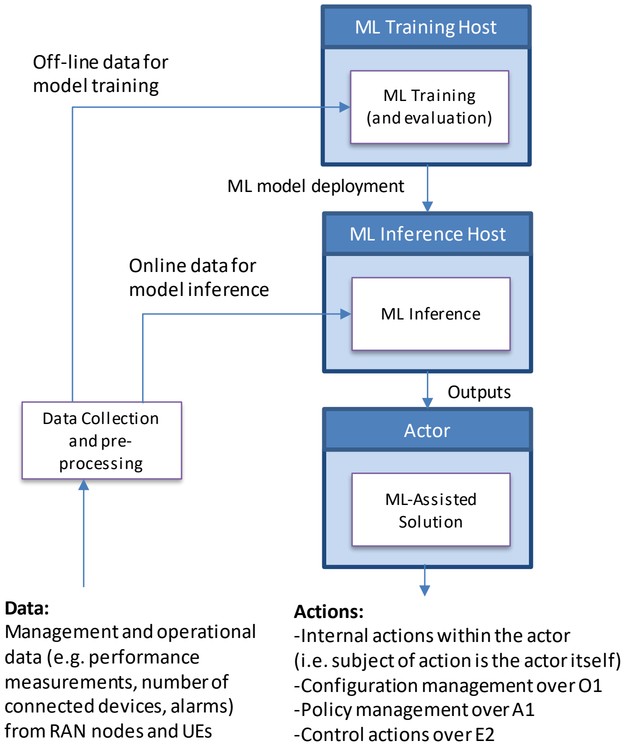

Fig. 2 – Components and relations for ML-assisted solutions within O-RAN

As shown in Fig. 2, a variety of management and operational data is collected from the different RAN nodes and User Equipment (UE) devices. Such data, properly preprocessed, is used to feed the two key components of the ML processing workflow, denoted as the ML training host and the ML inference host. The ML training host represents the runtime environment where offline training of the ML model takes places. This refers to the training of model before being executed within the network. In addition to the data collected from the real network, offline training may also rely on synthesized data which can accurately reproduce the behavior of the real network environment. The training may include an evaluation stage to assess the performance of the model and validate that it is ready and reliable to be deployed in the live network environment. Offline training is necessary to obtain supervised learning models (e.g. deep neural networks, support vector machines, etc.) as well as reinforcement learning models (e.g. Q-learning, multi-armed bandit learning, deep RL). The training host component is likely to be part of the SMO layer.

The ML inference host represents the runtime environment where the (previously trained and validated) ML model is executed and fed with online data to produce the outputs that will be used in the network operation. Multiple ML inference hosts can be in place, whose location depends on aspects such as the purpose and type of ML models being executed, its computation complexity, the availability and the quantity of data used and the response time requirements (real-time or non-real-time) of the ML application. Hence, ML inference hosts can be placed within the SMO layer but also within the RAN nodes (i.e. near-RT RIC, O-CU, O-DU).

In turn, the actor represents the network entity (i.e. UE, O-DU, O-DU, Near-RT RIC and Non-RT RIC) that hosts the decision-making function that consumes the outputs of the ML inference host and takes actions. It is worth noting that the distinction between the ML inference host and the actor obeys the fact that these components may or may not be co-located as part of the same network entity. An example of non-co-location could be the case of a mobility prediction model executed in an inference host within the non-RT RIC that produces outputs (e.g. mobility patterns) that are retrieved and consumed by the near-RT RIC (i.e. the actor in this case) for enhanced RRM (e.g. handover decisions based on mobility patterns). In contrast, an example of co-location could be an RRM algorithm for mobility management that embeds a reinforcement learning model and is executed within the near-RT RIC, which in this case serves as both the inference host and the actor. The actions decided by the actor can be handled either internally within the actor (e.g. RL-based RRM algorithm for mobility management within the near-RT RIC) or enforced into other network components via the different specified interfaces. For example, management configuration actions from an actor within the SMO layer on any RAN node can be conducted via the O1 interface, control actions on O-CU/O-RU from an actor within the near-RT RIC can go over the E2 interface and policy management configuration actions between the non-RT RIC and the near-RT RIC can be communicated over the A1 interface.

3.2 Information models for network slice management

With regard to the management of network slicing in 5G networks, 3GPP specifications include information model definitions, referred to as Network Resource Models (NRMs), for the characterization of network slices [42] together with a set of management services (MnS) for network slice life-cycle management (e.g. network slice provisioning MnS for network slice creation, modification and termination, performance monitoring services per slice, etc.) [43]. In addition, work is being conducted at 3GPP level to support SLA/SLS management [44], as well as closed loop assurance solutions that allow a service provider to continuously deliver the expected level of communication service quality in a 5G network [45].

Fig. 3 provides an overview of the different types of information models and their relations that are relevant for network slice management. The main idea behind the overall flow of the information models, as illustrated in Fig. 3, is that a network slice is conceived as a “product” offered by a Network Slice Provider (NSP) to a Network Slice Customer (NSC). In this respect, the GSMA Generic Slice Template (GST) is used as the SLA information associated with the network slice product for the communication between the NSC and NSP through, e.g. a Business Support Systems (BSS) product order management Application Programming Interface (API).

The GSMA GST provides a standardized list of attributes (e.g. performance related, function related, etc.) that can be used to characterize different types of network slices [46]. GST is generic and is not tied to any type of network slice or to any agreement between an NSC and an NSP. A Network Slice Type (NEST) is a GST filled with (ranges of) values. There are two kinds of NESTs: Standardized NESTs (S-NEST), i.e. NESTs with values established by standards organizations, working groups, fora, etc. such as, e.g. 3GPP, GSMA, 5GAA, 5G-ACIA, etc.; and Private NESTs (P-NEST), i.e. NESTs with values decided by the NSP. Among the attributes included in the GST there are:

- Attributes that specify the area where the terminals can access a particular network slice and the spectrum used.
- Attributes that define the services and capabilities supported in the slice (e.g. "support for non-IP traffic", "MMTel support", "NB-IoT support", "Positioning support", etc.)
- Attributes that establish the capacity and performance of the slice, including guaranteed and maximum data rates per slice and per UE, as well as maximum number of concurrent sessions and terminals in the slice.
- Attributes that define the terminal mobility conditions and density of terminals.
- Attributes that define management and operational aspects and features of the slice (e.g. performance monitoring indicators, performance prediction indicators, user management openness).
- Attributes that define the isolation level of the slice with regard to other slices (e.g. physical isolation, logical isolation).

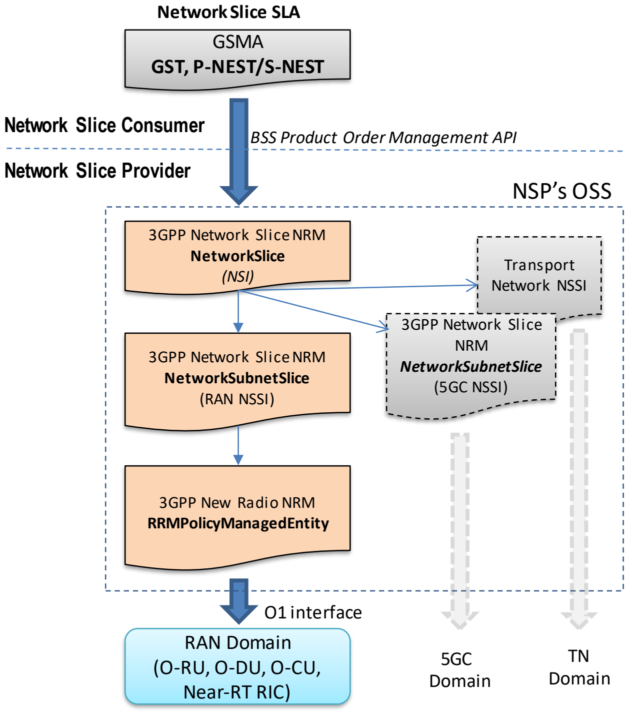

Fig. 3 – Information models for network slice management

Internally, within an NSP's OSS, the managerial representation of the network slice is realized with two Information Object Classes (IOC), named *NetworkSlice* and *NetworkSliceSubnet*, specified in the 3GPP information model definitions for "network slice NRM" [42]. 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. It is worth clarifying at this point that the realization of an NSI may be tied to the realization of several NSSIs, that is, an NSI that is composed of, e.g. a RAN NSSI, a 5G Core (5GC) NSSI and a Transport Network NSSI. However, depending on the NSP's product offering, it is also possible the realization of an NSI composed of a single domain such as, an NSI consisting of a single, RAN-only NSSI. Within the *NetworkSlice* IOC and *NetworkSubnetSlice* IOC, the attributes that are defined to encode the network slice related requirements that should be supported by the NSI and the NSSI are named, respectively, *ServiceProfile* and *SliceProfile*. Such attributes are compound data types that include attributes directly inherited from the GSMA GST template, as well as additional attributes to capture more specific requirements derived from the service performance requirements defined in [47], [48] . In particular, let us introduce here three attributes included in the *ServiceProfile* that are are directly inherited from the GSMA GST and used in the algorithm presented in Section 4:

- *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.

And last but not least, as also captured in Fig. 3, 3GPP also provides information model definitions for "New Radio NRM" [42], where different RAN management parameters are defined to configure the behaviour of the RAN nodes with regard to the operation of the established network slices. In this respect, a more detailed, though still simplified, view of the classes and attributes of the "New Radio NRM" model that allow for the characterization of the RRM policies for configuring the way that resources are allocated to the slices is provided in Fig. 4. Specifically, 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 and 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 Assistance Information (S-NSSAI) and Public Land Mobile Network Identifier (PLMNid).

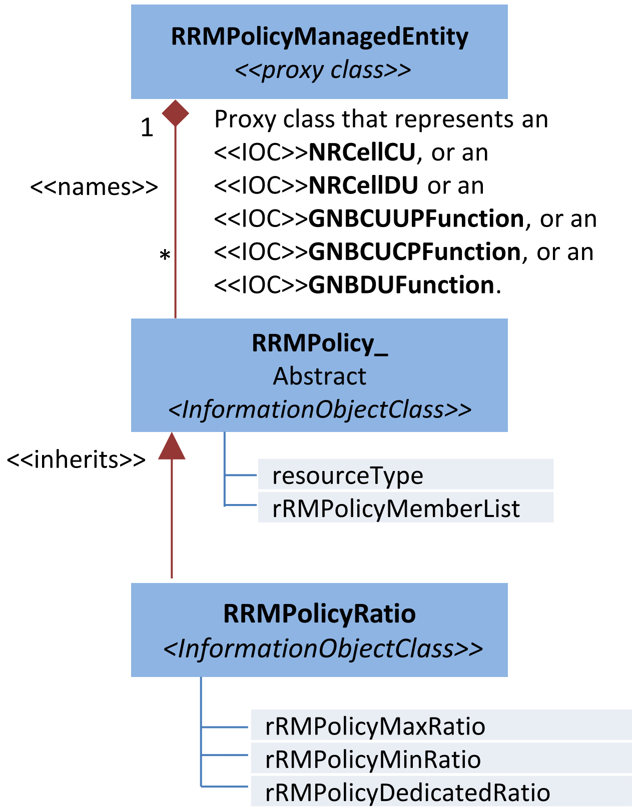
RRMPolicyManagedEntity
<<proxy class>>
1
<<names>>
*
Proxy class that represents an <<IOC>>NRCellCU, or an <<IOC>>NRCellDU or an <<IOC>>GNBCUUPFunction, or an <<IOC>>GNBCUCPFunction, or an <<IOC>>GNBDUFunction.
RRMPolicy_
Abstract
<InformationObjectClass>>
<<inherits>>
resourceType
rRMPolicyMemberList
RRMPolicyRatio
<InformationObjectClass>>
rRMPolicyMaxRatio
rRMPolicyMinRatio
rRMPolicyDedicatedRatio

Fig. 4 – Simplified representation of the classes and attributes for configuration of RAN functions subject to RRM policies for slice management.

On this basis, the *RRMPolicyRatio* IOC represents a particular realization of an RRMPolicy. Specifically, it 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.

4. ML-ASSISTED CROSS-SLICE OPTIMIZATION SOLUTION

This section describes an ML-assisted solution for cross-slice optimization based on the O-RAN framework and network slicing information models presented in the previous section. The solution is conceived to be deployed as part of the RAN SMO. The functional model and components of the solution are illustrated in Fig. 5. The cross-slice radio resource optimization problem considered here consists of dynamically adjusting the amount of downlink radio resources assigned to each RAN slice in each of the cells where the RAN slice is accessible in order to account for the spatio-temporal traffic variations across the cells. The solution is designed to operate with N cells and K RAN slices and keep track of the traffic variations in periods (time steps) of Δt minutes. This is achieved through the dynamic configuration of the *rRMPolicyDedicatedRatio* attribute of each cell on a per RAN slice basis. This configuration is conducted via the management provisioning services offered by the O1 interfaces, as seen in Fig. 5. 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.

For determining the *rRMPolicyDedicatedRatio* for the n-th cell and the k-th slice, denoted as $\sigma(k,n)$, the

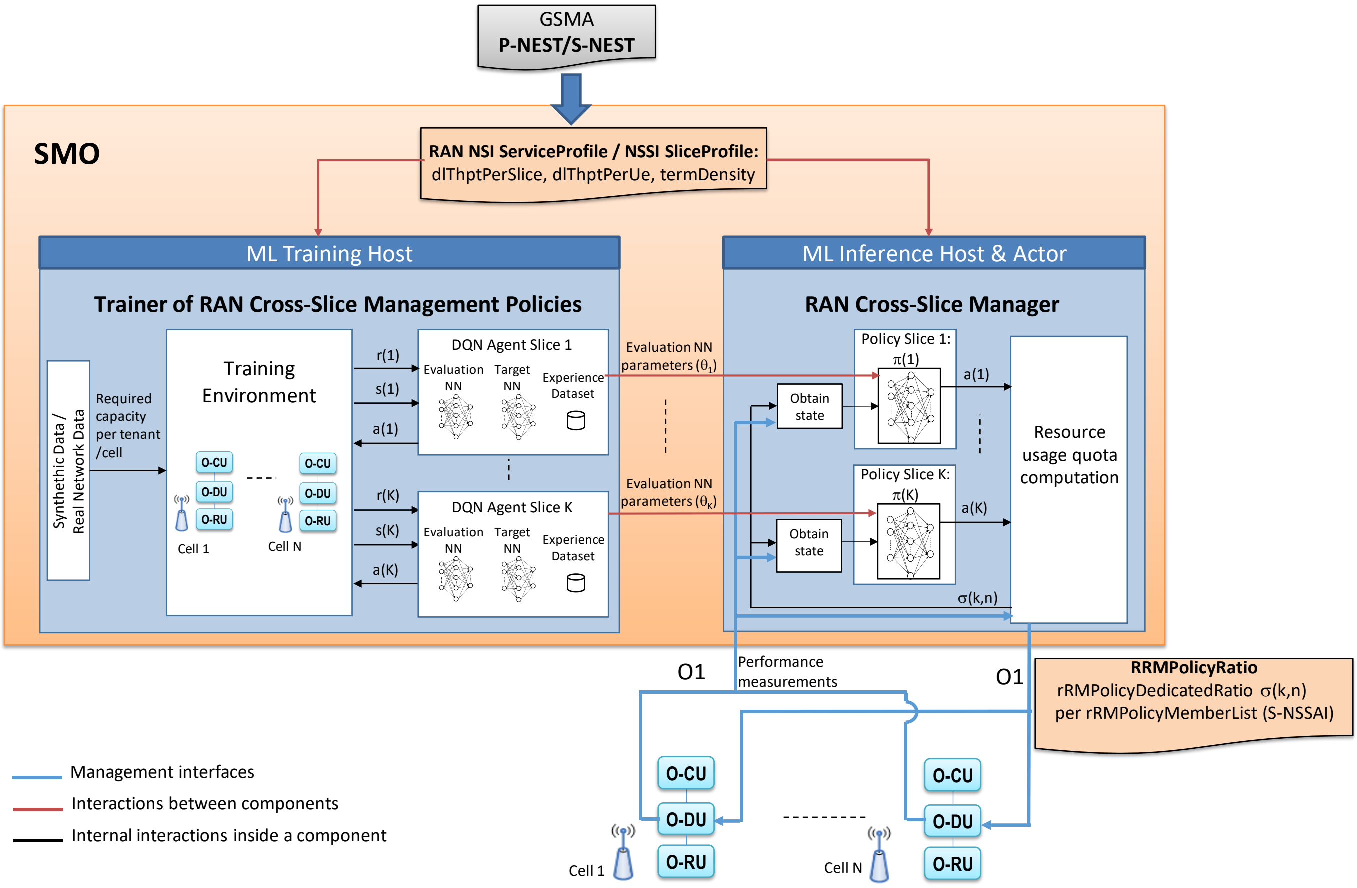
GSMA
P-NEST/S-NEST
SMO
RAN NSI ServiceProfile / NSSI SliceProfile:
dlThptPerSlice, dlThptPerUe, termDensity
ML Training Host
Trainer of RAN Cross-Slice Management Policies
Synthethic Data / Real Network Data
Required capacity per tenant /cell
Training Environment
O-CU
O-DU
O-RU
Cell 1
Cell N
r(1)
s(1)
a(1)
r(K)
s(K)
a(K)
DQN Agent Slice 1
DQN Agent Slice K
Evaluation NN
Target NN
Experience Dataset
Evaluation NN parameters (θ_1)
Evaluation NN parameters (θ_K)
ML Inference Host & Actor
RAN Cross-Slice Manager
Policy Slice 1: $\pi(1)$
Policy Slice K: $\pi(K)$
Obtain state
a(1)
a(K)
Resource usage quota computation
$\sigma(k,n)$
O1
Performance measurements
O1
RRMPolicyRatio
rRMPolicyDedicatedRatio $\sigma(k,n)$
per rRMPolicyMemberList (S-NSSAI)
Management interfaces
Interactions between components
Internal interactions inside a component
Cell 1
Cell N

Fig. 5 – Deep Q Network-based cross-slice optimization solution

proposed ML-based solution relies on Multi-Agent Reinforcement Learning (MARL) based on Deep Q-Network (DQN) whose mathematical details can be found in [33]. An important advantage of this multi-agent scheme is that it uses slice-specific DQN agents and action selection policies for the training and inference processes and, therefore, slices can be easily added/removed in the scenario just by adding or removing the corresponding agent and action selection policy. Specifically, as seen in Fig. 5, as part of the solution there is a Resource Usage Quota Computation module that determines the values of $\sigma(k,n)$ based on the outputs obtained through the execution of K action selection policies $\pi(k)$, each one associated to one slice. Each one of these policies is specified through a deep neural network (NN) defined by a vector of parameters θ_k that have been previously learnt during the training process, as illustrated in Fig. 5.

The solution considered here assumes that the SLA specification of the RAN slice requirements is done based on three ServiceProfile parameters explained in Section 3, namely *dlThptPerSlice*, *dlThptPerUe* and *termDensity*, which are directly derived from the GSMA GST template and used as inputs for the different solution components described in more detail in the following subsections. The specific values of these parameters for the slice k are denoted as *dlThptPerSlice*(k), *dlThptPerUe*(k) and *termDensity*(k) [1].

4.1 RAN cross-slice manager

This component includes the inference part of the DQN model and the functions needed to configure the RAN nodes through the O1 interface (i.e. it takes the ML inference and actor roles of the O-RAN

[1] For the interested reader, the parameters of the algorithm described in [33] are related to the considered Service Profile attributes as follows: Scenario Aggregated Guaranteed Bit Rate (SAGBR)=*dlThptPerSlice*; Maximum Cell Bit Rate (MCBR) of slice k in cell n: *MCBR*(k,n)=*dlThptPerUe*(k) × *termDensity*(k) × cell n service area.

framework). The computation of $\sigma(k,n)$ and potential update of the *rRMPolicyDedicatedRatio* attribute is done every Δt minutes. The determination of the resource usage quota $\sigma(k,n)$ is realized through the following functions:

- *Per-slice action selection policies*

The action selection policy of slice k gets the network state $\boldsymbol{s}(k)$ observed for this slice at the time when the policy is executed and determines the action $\boldsymbol{a}(k)$ to be applied for this slice. The action $\boldsymbol{a}(k)$ is composed of N per-cell actions that take one out of three possible values corresponding to: increase the resource usage quota $\sigma(k,n)$ for slice k in cell n in an amount of Δ for the next time step, maintain the same resource usage quota or decrease it in an amount of Δ.

In turn, the state $\boldsymbol{s}(k)$ includes N different per-cell components, each one given by the triple $<\rho(k,n),\sigma(k,n), \sigma_{ava}(n)>$ where $\rho(k,n)$ is the fraction of PRBs occupied by the slice k in cell n, $\sigma(k,n)$ is the current resource usage quota allocated to the slice and $\sigma_{ava}(n)$ is the total amount of resource usage quota in the cell not allocated to any slice. While the values of $\sigma(k,n)$ and $\sigma_{ava}(n)$ are directly available at the RAN cross-slice manager, the value of $\rho(k,n)$ is obtained from the different cells through the performance management (PM) services offered by the O1 interface. In particular, using the performance measurements defined in [49], it corresponds to the ratio between the "DL PRB used for data traffic", which measures the number of PRBs used in average for data traffic in a given slice and cell, and the "DL total available PRB", which measures the number of available PRBs in the cell. Both measurements are collected from the gNB-DU every time step, so their average is performed along the time step duration Δt.

Following the DQN approach, the action selection policy $\pi(k)$ of the k-th slice seeks to maximize a cumulative reward that captures the desired optimization target to be achieved. In particular, the action selection policy $\pi(k)$ for a given state $\boldsymbol{s}(k)$ is defined as $\text{argmax}_{\boldsymbol{a(k)}} Q_k(\boldsymbol{s}(k),\boldsymbol{a}(k),\theta_k)$, where $Q_k(\boldsymbol{s}(k),\boldsymbol{a}(k),\theta_k)$ is the output of a deep NN for the input state $\boldsymbol{s}(k)$ and the output action $\boldsymbol{a}(k)$, providing the maximum expected cumulative reward starting at $\boldsymbol{s}(k)$ and triggering $\boldsymbol{a}(k)$. The internal structure of the NN is specified by the vector of parameters θ_k that contains the weights of the different neuron connections. The optimum values of θ_k that determine the policies to be followed by the different slices in order to maximize the cumulative reward are learnt offline by the ML training host who provides them to the ML inference host. Further details about this training process and the reward formulation are given in Section 4.2.

- *Resource usage quota computation*

This function computes the value of the resource usage quota $\sigma(k,n)$ to be allocated to each slice and cell for the next time step by applying the increase/maintain/decrease actions provided by the action selection policies of all the slices and configures the resulting $\sigma(k,n)$ values in the O-DU through the *rRMPolicyDedicatedRatio* attribute. 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.

When applying the actions, this function ensures that the maximum cell bit rate value associated to the *termDensity* and *dlThptPerUe* parameters is not exceeded. Moreover, since the action selection policies for the different slices operate independently, this function also checks 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. If this happens, it applies first the actions of the slices involving a reduction or maintenance of the resource usage quota and the remaining capacity is distributed among the slices that have increase actions. This distribution is proportional to their *dlThptPerSlice* values, as long as their current throughput is not already higher than the *dlThptPerSlice*. For doing this adjustment, the measured throughput per slice across all the cells in the last time step is needed. It can be obtained from the PM services of the O1 interface using the "Downstream throughput for Single Network Slice Instance" Key Performance Indicator of [50].

4.2 Trainer of RAN cross-slice management policies

This component constitutes the training part of the DQN model intended to learn the NN parameters θ_k that determine the per-slice action selection policies to be used by the RAN cross-slice manager.

The training process makes use of a multi-agent DQN approach in which each DQN agent learns the optimum policy of a different RAN slice by continuously interacting with a training environment and updating the NN parameters as a result of these interactions. The training environment considered here is a network simulator that mimics the behavior of the real network when varying the offered load of the different slices in the different cells and when modifying the resource usage quota allocated to each slice as a result of the actions made by the DQN agents. In this respect, the simulator is fed by training data consisting of multiple time patterns of the required capacity (i.e. offered load) of the slices in the different cells. This data can be either built synthetically or extracted from real network measurements. The training is assumed to be executed in a training host, located at the SMO, with the necessary libraries, supporting tools and computational capabilities for training the DQN models and running the simulator.

For carrying out the training process, each DQN agent is composed by three different elements: (i) The evaluation NN, which corresponds to the $Q_k(\boldsymbol{s}(k),\boldsymbol{a}(k),\theta_k)$ being learnt that will eventually determine the policy to be applied at the ML inference host. (ii) The target NN, which is another NN with the same structure as the evaluation NN but with weights θ_k^-. It is used for obtaining the so-called Time Difference (TD) target required for updating the evaluation NN. (iii) The experience data set (ED), which stores the experiences of the agent resulting from the interactions with the training environment as explained in the following.

The interactions between the DQN agent and the training environment occur in time steps of (simulated time) duration Δt. In each time step the DQN agent of the k-th slice observes the state $\boldsymbol{s}(k)$ in the training environment and selects an action $\boldsymbol{a}(k)$. Action selection is based on an ε-Greedy policy that, with probability 1-ε, chooses the action that maximizes the output of the evaluation NN, and, with probability ε, chooses a random action. As a result of applying the selected action, the training environment generates a reward value $r(k)$ that assesses how good the action was from the perspective of the desired behavior. In particular, in the considered approach the reward captures both the SLA satisfaction and the capacity utilization. In this way, the reward for slice k is defined as the weighted product of three terms given by:

$$r(k) = \gamma_{SLA}(k)^{\varphi_1} \cdot \left(\frac{1}{K-1}\sum_{\substack{k'=1\\k'\neq k}}^{K} \gamma_{SLA}(k')\right)^{\varphi_2} \cdot \gamma_u(k)^{\varphi_3} \quad (1)$$

where φ_1, φ_2 and φ_3 are the weights of each component.

The first and second components in (1) correspond, respectively, to the SLA satisfaction ratio $\gamma_{SLA}(k)$ of the slice k and the aggregate for the rest of slices $k'\neq k$. Specifically, $\gamma_{SLA}(k)$ is the ratio between the aggregate throughput obtained by the slice across all cells $T(k)$ and the minimum between the aggregate offered load $A(k)$ and the $dlThptPerSlice(k)$ term of the SLA and is computed as:

$$\gamma_{SLA}(k) = min\left(\frac{T(k)}{min(dlThptPerSlice(k), A(k))}, 1\right) \quad (2)$$

where $A(k)$ is the aggregate across all the cells of the per-cell offered load $O(k,n)$ of slice k bounded by the limit established by the $TermDensity(k)$ and $dlThptPerUe(k)$ parameters of the SLA in the service area $S(n)$ of each cell n, that is:

$$A(k) = \sum_{n=1}^{N} min\big(O(k,n), dlThptPerUe(k) \cdot TermDensity(k) \cdot S(n)\big) \quad (3)$$

The third component of the reward is the capacity utilization factor, $\gamma_u(k)$, which aims at minimizing the over-provisioning of capacity and is defined as the ratio between the aggregate throughput $T(k)$ obtained by the slice and the total capacity allocated to the slice across all cells, that is:

$$\gamma_u(k) = \frac{T(k)}{\sum_{n=1}^{N} C(n)\cdot\sigma(k,n)} \quad (4)$$

where $C(n)$ is the capacity of cell n.

The reward $r(k)$ is provided by the training environment to the DQN agent at the end of each time step and, correspondingly, the $T(k)$ and $A(k)$ values correspond to average values during the time step.

As a result of the interactions between the training environment and the DQN agent, each experience of the ED is represented by a tuple that includes the state observed at the beginning of a given time step, the selected action, the obtained reward as a result of this action and the new state observed at the end of the time step duration.

The experiences stored in the ED are used by the DQN agent to progressively update the values of the weights θ_k and θ_k^- in the evaluation and target NNs, respectively. For each time step, the update of the weights θ_k of the evaluation NN is performed by randomly selecting a mini batch of experiences of the ED and updating the weights of the evaluation NN θ_k according to the mini-batch gradient descent method. Moreover, the weights θ_{k^-} of the target NN are updated with the weights of the evaluation NN every M time steps. The reader is referred to [33] for details on the mathematical formulation of this process.

The training process stops after a sufficient number of time steps that ensures the convergence of the process. At this point, the ML training host is ready to provide the evaluation NN parameters θ_k so that the model can be applied on the real network using the ML inference host.

5. ILLUSTRATIVE SCENARIO AND EVALUATION

To illustrate the behavior the proposed cross-slice optimization framework and ML-assisted solution, let us consider a scenario with a localized RAN deployment run by an infrastructure provider, serving as an NSP, which offers a RAN slice product to a pair of MNOs, which in this case act as NSCs. This could be the case of a dense urban deployment of small cells in a business district operated under a neutral host model. Let us assume that the MNOs use the RAN slices to offer enhanced Mobile BroadBand (eMBB) services to their customers so that key parameters to include in the SLA are the number of UEs expected to be served in the area, given in terms of the maximum terminal density, the throughput guaranteed in the whole service area per slice and the expected UE experienced data rates. These SLA parameters are summarized in Table 1. On the other hand, let us assume a RAN deployment consisting of 5 small cells, which provide an aggregated capacity of 10 Gb/s in an area of 0.15 km^2. The characteristics of this deployment are captured in Table 2. Under such settings, note that the *dlThptPerSlice* values of MNO1 and MNO2 SLAs actually account for the 60% and 40% of the total capacity, respectively.

Two different cases of offered load patterns of the MNOs throughout the day are considered for evaluating the performance of the learnt policies. Case 1 shown in Fig. 6 corresponds to a situation in which the offered loads of the two MNOs exhibit a certain complementarity during the time period comprised between 900 and 1300 min, approximately, in which MNO2 exhibits a large load while the load of MNO1 is kept at low values. Instead, Case 2 shown in Fig. 7 reflects a situation in which the offered load of the two MNOs is more correlated. In both Fig. 6 and Fig. 7 the offered load corresponds to a period of one day and is represented as the average in intervals of 15 min. It is worth mentioning that the focus of the results is put on the temporal variations of the offered load so, from the spatial perspective, it is assumed, for simplicity, that the aggregate offered load is homogeneously distributed across the cells.

Table 1 – SLA parameters

GSM GST Attributes	MNO1	MNO2
dlThptPerSlice	6 Gb/s	4 Gb/s
termDensity	1000 UEs/km^2	500 UEs/km^2
dlThptPerUe	50 Mb/s	100 Mb/s

Table 2 – Cell configuration

Parameter	Value
Number of cells	5
Cell radius	100m
Cell bandwidth	100 MHz (273 PRBs with 30 kHz subcarrier spacing)
Average spectral efficiency	5.1 b/s/Hz
MIMO configuration	Spatial multiplexing with 4 layers
Total cell capacity	2 Gb/s

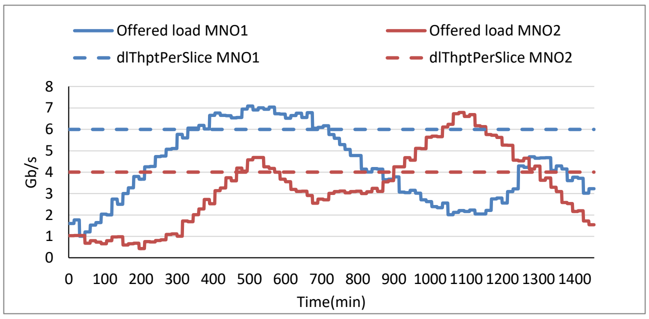

Fig. 6 – Offered load pattern of each MNO in Case 1.

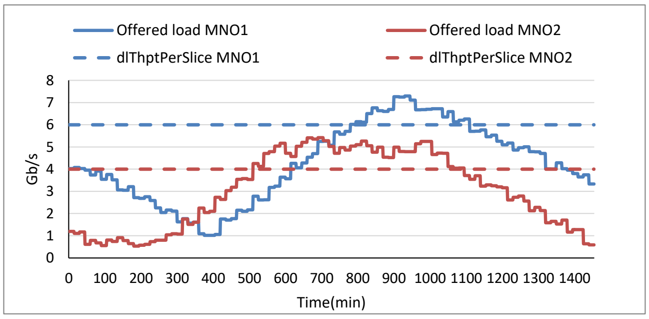

Fig. 7 – Offered load pattern of each MNO in Case 2.

The DQN-based cross-slicing solution has been implemented in Python by using the library *TF-Agents* [51]. Table 3 shows the parameters of the DQN model (see [33] for details on these parameters). To obtain the values of these parameters a prior analysis of the model behavior with different combinations of parameters has been conducted. The model has been trained using a data set composed of 140 synthetically generated offered load patterns of the two MNOs in the different cells during one day. They capture different load levels and situations of complementarity among MNOs, in order that the DQN agents can visit multiple states during the training process.

Table 3 – DQN model parameters

Parameter	Value
Initial collect steps	5000
Number of training steps	10^6
Experience Data set maximum length	10^7
Mini-batch size	256
Learning rate	0.0001
Time steps between updates of the target NN weights (M)	1
Discount factor	0.9
ε value (ε-Greedy)	0.1
Neural network nodes	2 layers of 100 nodes
Resource quota increase (Δ)	0.1
Time step duration (Δt)	1 min
Reward weights (φ_1, φ_2, φ_3)	(0.3, 0.2, 0.5)

The training has been conducted with a system level network simulator that considers the offered load patterns of the different slices and cells as input. In every time step the DQN agents select the actions that determine the *rRMPolicyDedicatedRatio* assigned to each slice in each cell. Then, the number of PRBs that are utilized by the slice is the minimum between the assigned PRBs in accordance with *rRMPolicyDedicatedRatio* and the required PRBs, which are determined by the offered load and the spectral efficiency. Then, the throughput achieved by each slice is obtained using the number of utilized PRBs and the spectral efficiency. From this, the SLA satisfaction ratio from (2), the capacity utilization from (4) and the reward from (1) are computed. The reward, together with the selected action and the actual and previous states are stored in the experience data set and the weights of the evaluation and target NNs are updated. This process is repeated until reaching the number of training steps indicated in Table 3. At the end, the resulting weights of the evaluation NN determine the trained policy to be used during the ML inference stage.

Once the training has been completed, the ML inference stage assesses the obtained policy using the same system level network simulator of the training, but now taking as input the offered load patterns of Fig. 6 and Fig. 7 split equally among the different cells. The trained policy is executed every time step to obtain the *rRMPolicyDedicatedRatio* values, from which the SLA satisfaction ratio and capacity utilization metrics are determined.

To illustrate the operation of the considered cross-slicing solution, Fig. 8 and Fig. 9 plot the evolution of the *rRMPolicyDedicatedRatio* parameter in % configured by the algorithm for each slice in one of the cells for Cases 1 and 2, respectively. As a reference, the evolution of the offered load pattern of each MNO, measured in % of the total scenario capacity is also shown in the plots.

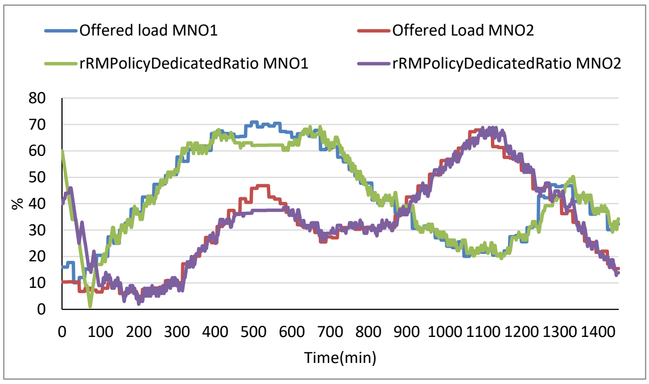

Fig. 8 – Evolution of the *rRMPolicyDedicatedRatio* for each MNO in one cell for Case 1.

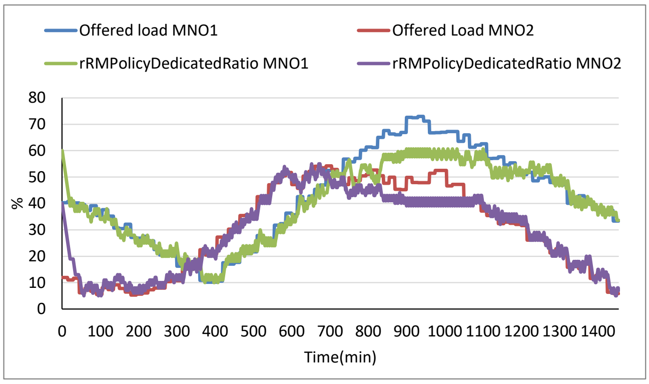

Fig. 9 – Evolution of the *rRMPolicyDedicatedRatio* for each MNO in one cell for Case 2.

Focusing on Fig. 8, it can be observed that in general the algorithm is able to modify the amount of resources assigned to each slice through the *rRMPolicyDedicatedRatio* parameter following the offered load fluctuations, so that the algorithm provides each slice with the resources it needs to support its load. Going into further details, different situations can be identified during the time evolution of Fig. 8.

Initially at time *t*=0 min the *rRMPolicyDedicatedRatio* is set to 60% and 40% for slice 1 and slice 2. These values correspond to the fractions of resources associated to the *dlThptPerSlice* values established in the SLA. Then, as time increases, an initial transient period of around 100 min is observed in which the *rRMPolicyDedicatedRatio* is progressively adjusted to fit the actual resource demand. After this period, the *rRMPolicyDedicatedRatio* approximately follows the offered load of each MNO in the cell, as long as the aggregate load for all cells is below the *dlThptPerSlice* values.

A particular situation occurs between *t*=350 min and *t*=450 min, and between *t*=600 and 700 min, approximately. In these periods the offered load of MNO1 exceeds its *dlThptPerSlice* value of the SLA (60%). At the same time, the offered load of MNO2 during this period is still below its corresponding *dlThptPerSlice* value (40%) and the cell has enough resources to satisfy the demands of both MNOs. Therefore, for the sake of better resource utilisation, the algorithm allows the *rRMPolicyDedicatedRatio* of the slice of MNO1 to exceed the 60%, providing in this way all the required capacity. An equivalent situation occurs between time *t*=900 min and *t*=1300 min, but now it is the offered load of MNO2 that clearly exceeds its *dlThptPerSlice* while the load of MNO1 is substantially lower than its *dlThptPerSlice*. Again, since the cell has enough capacity to satisfy the demands of both MNOs, the *rRMPolicyDedicatedRatio* of the slice of MNO2 is increased beyond the *dlThptPerSlice* value of 40% to support all the load of this MNO.

In turn, the period between *t*=450 min and *t*=600 min corresponds to the case in which both MNOs are demanding capacity beyond their *dlThptPerSlice* values and the cell does not have sufficient resources to support all the demand (i.e. the sum of the offered loads of MNO1 and MNO2 exceeds 100%). For this reason, it is observed in Fig. 8 that the algorithm sets the *rRMPolicyDedicatedRatio* in accordance with the *dlThptPerSlice* values, i.e. 60% for MNO1 and 40% for MNO2.

Focusing now on Case 2, which reflects a larger correlation between the offered load of both MNOs, the results of Fig. 9 reveal a similar behavior than the one discussed in Fig. 8, and the algorithm is able to allocate to each slice the necessary amount of resources to support their offered load. This occurs even when the offered load of a slice is above the *dlThptPerSlice* limit as long as there are sufficient resources in the cell. In turn, when the load of both MNOs exceeds the *dlThptPerSlice* value and there are not enough resources in the cell (e.g. between *t*=850 and *t*=1150 min in Fig. 9) the algorithm sets the *rRMPolicyDedicatedRatio* in accordance with the *dlThptPerSlice* values.

Table 4 presents some indicators to quantitatively assess the performance of the cross-slicing approach. Specifically, the average SLA satisfaction obtained for each MNO is presented. This is measured as the time average of equation (2) over the whole simulation time of 1440 min (excluding the initial transient period of 100 min) and provides the percentage of time in which the *rRMPolicyDedicatedRatio* has allocated enough resources to support the offered load while this load was below the *dlThptPerSlice* value established in the SLA. It is observed that the algorithm achieves high SLA satisfaction, above 97% in all the cases. To further quantify the variability of the SLA satisfaction, the 5th percentile of this indicator is also shown in the table. The large obtained values around 91% reflect that the achieved SLA satisfaction is very good most of the time.

Similarly, to account for the degree of resource over-provisioning when allocating resources to each slice (i.e. for the situations in which the *rRMPolicyDedicatedRatio* allocated to a slice includes more resources than actually required by the MNO), Table 4 also shows the average capacity utilization of the assigned resources. This is measured as the time average of the ratio between the throughput of a slice and the amount of allocated capacity to this slice from equation (4). The average is measured along the whole simulation time of 1440 min excluding the initial transient period of 100 min. The corresponding 5th percentile is also indicated in Table 4. It is observed that the algorithm achieves high utilization, being the average above 94% and the 5th percentile approximately above 80% for both slices. This reflects that the algorithm is able to properly adjust the resource allocation to the actual needs and thus to reduce over-provisioning situations.

Table 4 – Performance indicators

Performance indicator	**Case 1**		**Case 2**	
	MNO1	**MNO2**	**MNO1**	**MNO2**
SLA satisfaction (average)	98.31%	97.69%	97.38%	97.14%
SLA satisfaction (perc. 5)	92.27%	90.97%	91.95%	91.75%
Capacity utilization (average)	96.44%	94.06%	96.43%	94.61%
Capacity utilization (perc. 5)	90.18%	81.79%	90.33%	79.28%

6. CONCLUDING REMARKS

ML-assisted solutions with the ability to learn particular network dynamics and operate under goal-oriented policies arise as a feasible approach to tackling the complexity of the cross-slice radio resource allocation problem. Beyond the algorithmic dimension, bringing into the equation the information models and the architectural context for the implementation of these ML-assisted solutions is also key to further progress towards their practical realization.

In this respect, this paper has outlined the building blocks of the architectural framework being established under the O-RAN Alliance for the deployment of ML-assisted solutions in the RAN along with the different types of information models developed by 3GPP for network slice management, from service characterization at the SLA level down to the specific management attributes that can be used to configure how radio resources are allocated to the slices within the RAN nodes.

Building upon this architectural framework and associated information models, the paper has presented a plausible realization of an ML-assisted cross-slice radio resource optimization solution based on the use of multi-agent DQN techniques. The presented solution is shown to be a feasible approach to dynamically adjust the resource allocation of the slices to the traffic variations in order to fulfill an SLA and achieve high resource utilization efficiency.

ACKNOWLEDGEMENT

This work has been supported by the Spanish Research Council and FEDER funds under SONAR 5G grant (ref. TEC2017-82651-R) and by the Secretariat for Universities and Research of the Ministry of Business and Knowledge of the Government of Catalonia under grant 2019FI_B1 00102.

REFERENCES

[1] P. Rost, A. Banchs, I. Berberana, M. Breitbach, M. Doll, H. Droste, C. Mannweiler, M.A. Puente, K. Samdanis, B. Sayadi, “Mobile network architecture evolution toward 5G”, *IEEE Communications Magazine*, Vol. 54, No. 5, May, 2016, pp. 84-91.

[2] 3GPP TS 23.501 v16.6.0 “System Architecture for the 5G System; Stage 2 (Release 16)”, September, 2020.

[3] RP-201612, “Study on enhancement of RAN Slicing for NR”, 3GPP TSG RAN Meeting #89e, September, 2020.

[4] SP-190931, “Feasibility Study on Enhancement of Network Slicing Phase 2”, 3GPP TSG-SA Meeting #85, September, 2019.

[5] SP-200766, “Study on network slice management enhancement”, 3GPP TSG-SA Meeting #89e, September, 2020.

[6] SP-200571, “Feasibility Study on Enhanced Access to and Support of Network Slice”, TSG SA Meeting #88e, July, 2020.

[7] R. Ferrús, O. Sallent, J. Pérez-Romero, R. Agustí, “On 5G Radio Access Network Slicing: Radio Interface Protocol Features and Configuration”, *IEEE Communications Magazine*, Vol. 56, No. 5, May, 2018, pp.184-192.

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

[9] O. Sallent, J. Pérez-Romero, R. Ferrús, R. Agustí, “On Radio Access Network Slicing from a Radio Resource Management Perspective”, *IEEE Wireless Communications*, Vol. 24. No.5, October, 2017,pp. 166-174.

[10] A. S. D. Alfoudi, S. H. S. Newaz, A. Otebolaku, G. M. Lee and R. Pereira, "An Efficient Resource Management Mechanism for Network Slicing in a LTE Network”, *IEEE Access*, Vol. 7, July, 2019, pp. 89441-89457.

[11] 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, December, 2018, pp. 970-973.

[12] I. Vilà, J. Perez-Romero, O. Sallent, A. Umbert, “Characterization of Radio Access Network Slicing Scenarios with 5G QoS Provisioning”, *IEEE Access*, Vol. 6, March, 2020, pp. 51414-41430.

[13] D. Marabissi, R. Fantacci, "Highly Flexible RAN Slicing Approach to Manage Isolation, Priority, Efficiency”, *IEEE Access*, Vol. 7, July, 2019, pp. 97130-97142.

[14] R. Ferrus, O. Sallent, J. Perez-Romero, R. Agustí, "On the automation of RAN slicing provisioning: solution framework and applicability examples", *EURASIP Journal on Wireless Communications and Networking*, June, 2019

[15] 3GPP TR 28.861 v16.0.0 "Study on the Self-Organizing Networks (SON) for 5G networks (Release 16)", December, 2019.

[16] X. Shen, J. Gao, W. Wu, K. Lyu, M. Li, W. Zhuang, X. Li, J. Rao, "AI-Assisted Network-Slicing Based Next-Generation Wireless Networks", *IEEE Open Journal of Vehicular Technology*, Vol. 1, January, 2020, pp. 45-66.

[17] D. Bega, M. Gramaglia, A. Banchs, V. Sciancalepore, K. Samdanis, X. Costa-Perez, "Optimising 5G infrastructure markets: The Business of Network Slicing", *IEEE INFOCOM,* 2017.

[18] D. Bega, M. Gramaglia, A. Banchs, V. Sciancalepore, K. Samdanis, X. Costa-Perez, "A Machine Learning approach to 5G Infrastructure Market optimization", *IEEE Transactions on Mobile Computing,* Vol. 19, No. 3, March, 2020, pp. 498-512.

[19] N. V. Huynh, D. T. Hoang, D. N. Nguyen, E. Dutkiewicz, "Optimal and Fast Real-Time Resource Slicing with Deep Dueling Neural Networks", *IEEE Journal on Selected Areas in Communications*, Vol. 37, No. 6, June, 2019, pp. 1455-1470.

[20] V. Sciancalepore, K. Samdanis, X. Costa-Perez, D. Bega, M. Gramaglia, A. Banchs, "Mobile Traffic Forecasting for Maximizing 5G Network Slicing Resource Utilization", *IEEE INFOCOM*, 2017.

[21] V. Sciancalepore, X. Costa-Perez, A. Banchs, "RL-NSB: Reinforcement Learning-Based 5G Network Slice Broker", *IEEE/ACM Transactions on Networking*, Vol. 27, No.4, August, 2019, pp. 1543-1557.

[22] M. Toscano, F. Grunwald, M. Richart, J. Baliosian, E. Grampin, A. Castro, "Machine Learning Aided Network Slicing", *21st Int. Conf. on Transparent Optical Networks (ICTON)*, 2019.

[23] A. Aijaz, "Hap-SliceR: A Radio Resource Slicing Framework for 5G Networks with Haptic Communications", *IEEE Systems Journal*, Vol. 12, No. 3, September, 2018, pp. 2285-2296.

[24] B. Han, A. De Domenico, G. Dandachi, A. Drosou, D. Tzovaras, R. Querio, F. Moggio, O. Bulakci, H.D. Schotten, "Admission and Congestion Control for 5G Network Slicing", *IEEE Conf. on Standards for Communications and Networking (CSCN)*, 2018.

[25] X. Foukas, M.K.Marina, K. Kontovasilis, "Iris: Deep Reinforcement Learning Driven Shared Spectrum Access Architecture for Neutral-Host Small Cells", *IEEE Journal on Selected Areas in Communications*, Vol. 37, No. 8, August, 2019, pp. 1820-1837.

[26] S. D'Oro, L. Galluccio, P. Mertikopoulos, G. Morabito, S. Palazzo, "Auction-based resource allocation in OpenFlow multi-tenant networks", *Computer Networks*, Vol. 115, March, 2017, pp. 29-41.

[27] X. Chen, Z. Zhao, C. Wu, M. Bennis, H. Liu, Y. Ji, H. Zhang, "Multi-Tenant Cross-Slice Resource Orchestration: A Deep Reinforcement Learning Approach", *IEEE Journal on Selected Areas in Communications*, Vol. 37, No. 10, October, 2019, pp. 2377-2392.

[28] G. Sun, Z. T. Gebrekidan, G. O. Boateng, D. Ayepah-Mensah, W. Jiang, "Dynamic Reservation and Deep Reinforcement Learning Based Autonomous Resource Slicing for Virtualized Radio Access Networks", *IEEE Access*, Vol. 7, April, 2019, pp. 45758-45772.

[29] G. Sun, K. Xiong, G.O. Boateng, D. Ayepah-Mensah, G. Liu, W. Jiang, "Autonomous Resource Provisioning and Resource Customization for Mixed Traffics in Virtualized Radio Access Network", *IEEE Systems Journal*, Vol. 13, No. 3, September, 2019, pp. 2454-2465.

[30] R. Li, Z. Zhao, Q. Sun, C-L. I, C. Yang, X. Chen, M. Zhao, H. Zhang, "Deep Reinforcement Learning for Resource Management in Network Slicing", *IEEE Access*, Vol. 6, November, 2018, pp. 74429-74441.

[31] C. Qi, Y. Hua, R. Li, Z. Zhao, H. Zhang, "Deep Reinforcement Learning With Discrete Normalized Advantage Functions for Resource Management in Network Slicing", *IEEE Communications Letters*, Vol. 23, No. 8, August, 2019, pp. 1337-1341.

[32] Y. Abiko, T. Saito, D. Ikeda, K. Ohta, T. Mizuno, H. Mineno, "Flexible Resource Block Allocation to Multiple Slices for Radio Access

Network Slicing Using Deep Reinforcement Learning", *IEEE Access*, Vol. 8, April, 2020, pp. 68183-68198.

[33] I. Vilà, J. Pérez-Romero, O. Sallent, A. Umbert, "A Novel Approach for Dynamic Capacity Sharing in Multi-tenant Scenarios", *IEEE International Symposium on Personal, Indoor and Mobile Radio Communications (PIMRC)*, September, 2020.

[34] M. Yan, G. Feng, J. Zhou, Y. Sun and Y. Liang, "Intelligent Resource Scheduling for 5G Radio Access Network Slicing", *IEEE Transactions on Vehicular Technology*, Vol. 68, No. 8, August, 2019, pp. 7691-7703.

[35] D. Bega, M. Gramaglia, M. Fiore, A. Banchs, X. Costa-Perez, "DeepCog: Optimizing Resource Provisioning in Network Slicing with AI-based Capacity Forecasting", *IEEE Journal on Selected Areas in Communications*, Vol. 38, No.2, February, 2020, pp. 361-376.

[36] R. Giu, J. zhang, "GANSlicing: A GAN-Based Software Defined Mobile Network Slicing Scheme for IoT Applications", *International Conference on Communications (ICC)*, 2019.

[37] S. Khatibi, A. Jano, "Elastic Slice-Aware Radio Resource Management with AI-Traffic Prediction", *European Conference on Networks and Communications (EuCNC)*, 2019.

[38] O-RAN Alliance, "Operator Defined Next Generation RAN Architecture and Interfaces", https://www.o-ran.org, Accessed December, 2020.

[39] O-RAN Alliance, "O-RAN Use Cases and Deployment Scenarios: Towards Open and Smart RAN", white paper, February, 2020.

[40] O-RAN Alliance, "O-RAN A1 interface: General Aspects and Principles Version 1.0", October, 2019.

[41] 3GPP TS 28.530 v16.0.0, "Management and Orchestration; Concepts, use cases and requirements (Release 16)", September, 2019.

[42] 3GPP TS 28.541 V16.5.0, "Management and orchestration; 5G Network Resource Model (NRM); Stage 2 and stage 3 (Release 16)", June, 2020.

[43] 3GPP TS 28.531 V16.6.0, "Management and orchestration; Provisioning; (Release 16)", July, 2020.

[44] SP-200190, "New WID Enhancement on Management Aspects of 5G Service-Level Agreement", 3GPP TSG-SA Meeting #87e, March, 2020.

[45] SP-200196, "New WID on Enhanced Closed loop SLS Assurance", 3GPP TSG-SA Meeting #87e, March, 2020.

[46] GSMA NG.116 - Generic Network Slice Template Version 2.0 (2019-10-16).

[47] 3GPP TS 22.261 v17.3.0, "Service Requirements for the 5G System; Stage 1 (Release 17)", July, 2020.

[48] 3GPP TS 22.104 v17.3.0, "Service requirements for cyber-physical control applications in vertical domains; Stage 1 (Release 17)", July, 2020.

[49] 3GPP TS 28.552 v16.6.0, "Management and Orchestration; 5G performance measurements (Release 16)", July, 2020.

[50] 3GPP TS 28.554 v16.5.0, "Management and Orchestration; 5G end to end Key Performance Indicators (KPI) (Release 16)", July, 2020.

[51] S. Guadarrama, et. al. "TF-Agents: A library for Reinforcement learning in TensorFlow.", 2018. Available at: https://github.com/tensorflow/agents

AUTHORS

Ramon Ferrús received the degrees of Telecommunications Engineering (B.S. plus M.S.) and Ph.D. from the Universitat Politècnica de Catalunya (UPC), Barcelona, Spain, in 1996 and 2000, respectively. He is currently a tenured Associate Professor with the Department of Signal Theory and Communications at UPC. His research interests include system design, functional architectures, protocols, resource optimization and network and service management in wireless communications. He has participated in 10+ research projects within the 6th, 7th and H2020 Framework Programmes of the European Commission, taking the responsibility of WP leader in H2020 VITAL and FP7 ISITEP projects. He has also participated in numerous national research projects and technology transfer projects for public and private companies. He has participated in ETSI standardisation activities. He is co-author of one book on mobile and one book on mobile broadband public safety communications. He has co-authored over 120 papers mostly in IEEE journals and conferences, with a h-index of 23 in Google Scholar.

Jordi Pérez-Romero. is a professor in the Dept. of Signal Theory and Communications of the Universitat Politècnica de Catalunya (UPC) in Barcelona, Spain, where he received a degree in telecommunications engineering in 1997 and a Ph.D. degree in 2001. He has been working in the field of wireless communication systems, with a particular focus on radio resource management, cognitive radio networks and network optimization. He has been involved in different European projects with different responsibilities, such as researcher, work package leader, and Project Lead, has participated in different projects for private companies and has contributed to the 3GPP and ETSI standardization bodies. He has published more than 250 papers in international journals and conferences, three books and has contributed to seven book chapters. He has an h-index of 31 in Google Scholar. He serves as an Associate Editor for IEEE Vehicular Technology Magazine and EURASIP Journal on Wireless Communications Networks.

Oriol Sallent is a Professor at the Universitat Politècnica de Catalunya (UPC) in Barcelona. He has participated in a wide range of European and national projects, with diverse responsibilities as Principal Investigator, Coordinator and Workpackage Leader. He regularly serves as a consultant for a number of private companies. He has been involved in the organization of many different scientific activities, such as Conferences, Workshops, Special Issues in renowned international journals, etc. He has contributed to standardization bodies such as 3GPP, IEEE and ETSI. He is co-author of 13 books and has published 250+ papers, mostly in high-impact IEEE journals and renowed international conferences. His research interests include 5G RAN (Radio Access Network) planning and management, artificial intelligence-based radio resource management, virtualization of wireless networks, cognitive management in cognitive radio networks and dynamic spectrum access and management among others.

Irene Vilà received her B.E. degree in Telecommunication Systems Engineering and her M.E. degree in Telecommunication Engineering from the Universitat Politècnica de Catalunya (UPC), Barcelona, in 2015 and 2017, respectively. In 2018, she joined the Mobile Communication Research Group (GRCM) of the Department of Signal Theory and Communications (TSC) at UPC where she is currently a PhD student, supported with an FI AGAUR grant by the Government of Catalunya. Her current research interests include Radio Access Network (RAN) Slicing, network virtualization and the application of artificial intelligence and, particularly, machine learning to radio resource management.

Ramón Agustí received a degree of Engineer of Telecommunications from the Universidad Politécnica de Madrid, Spain, in 1973, and a Ph.D. degree from the Universitat Politècnica de Catalunya (UPC), Spain, 1978. He became Full Professor of the Department of Signal Theory and Communications (UPC) in 1987. After graduation he was working in the field of digital communications with particular emphasis on transmission and development aspects in fixed digital radio, both radio relay and mobile communications. For the last fifteen years he has been mainly concerned with aspects related to radio resource management in mobile communications. He has published about two hundred papers in these areas and co-authored three books. He participated in the European program COST 231 and in the COST 259 as Spanish representative delegate. He has also participated in the RACE, ACTS and IST European research programs as well as in many private and public funded projects. He received the Catalonia Engineer of the year prize in 1998 and the Narcis Monturiol Medal issued by the Government of Catalonia in 2002 for his research contributions to the mobile communications field. He is a Member of the Spanish Engineering Academy.