Towards Closed-loop Automation in 5G Open RAN: Coupling an Open-Source Simulator with xApps

Theofanis Karamplias[a], Sotirios T. Spantideas[a,*], Anastasios E. Giannopoulos[a], Panagiotis Gkonis[b], Nikolaos Kapsalis[a] and Panagiotis Trakadas[a]

[a] National and Kapodistrian University of Athens, Department of Ports Management and Shipping, Evia, 34400, Greece, email: faniskaramplias@gmail.com, sospanti@uoa.gr, angianno@uoa.gr, ncapsalis@pms.uoa.gr, ptrakadas@uoa.gr

[b] National and Kapodistrian University of Athens, Department of Digital Industry Technologies, Evia, 34400, Greece, email: pgkonis@uoa.gr

**Corresponding Author: Sotirios T. Spantideas (sospanti@uoa.gr)*

***Abstract*—The goal of this paper is to demonstrate the implementation of technological solutions that will enable the optimization of 5G network resources and services in an automated and self-configured manner. At first, the practical implementation of intelligence loops in the 5G network architecture is presented, according to the O-RAN specifications. Then, the development of an open source, general-purpose simulator, compliant with 3GPP specifications for generating physical-layer measurement reports from the radio access network is presented, while its functional logic and configuration capabilities are fully highlighted. Moreover, this paper illustrates how effectively trained machine learning (ML) models can be incorporated into the architecture for network configuration and optimization. In this context, an indicative use case is presented and evaluated, focusing on closed-loop power adjustment of the transmitters in a 5G cellular orientation, via the appropriate deployment of a deep reinforcement learning agent. The simulation results outline the interaction loop between the developed 5G simulator and the deployed ML model, targeting at increasing the network-wide throughput of user equipment.**

Keywords—5G; Open-source 5G Simulator; O-RAN; Machine Learning; Radio Intelligent Controller;

I. Introduction

The full deployment of fifth-generation (5G) broadband wireless cellular networks has accelerated the support of latency and bandwidth-demanding applications, as well the massive connectivity of heterogeneous devices [1]-[3]. In this context, various novel technologies have been introduced, such as network function virtualization (NFV), decoupling network node functionality from hardware equipment [4], [5]. Hence, a holistic infrastructure redesign can be supported according to traffic needs, where a large number of access points (APs) is deployed over small geographical regions (ultra-dense networking [6]). To this end, the exploitation of millimeter-wave (mmWave) frequency bands in the physical layer, along with the deployment of massive multiple-input multiple-output (m-MIMO) architectures over various APs, can leverage the provision of acceptable quality of service (QoS) over a mass number of user equipment (UEs) [7].

It becomes apparent from the above that 5G networks can be more complex in nature compared to the previous generations of mobile networks, due to the incorporation of diverse technologies and the associated signaling burden. Therefore, a constant monitoring and potential redesign is required in order to ensure their proper functionality. To this end, machine learning (ML) algorithms, being a subcategory of artificial intelligence (AI), can facilitate the optimum network design and deployment according to the processed monitoring data. Hence, many network operators and service providers are interested in the incorporation of the appropriate AI/ML algorithms in order to build networks that are capable of self-management and self-healing. ML-based applications can respond in near real-time data generated by the 5G network, thus providing new opportunities for process automation. In this context, the need for interoperability among various network operators and hardware components has leveraged the adoption of open access solutions for radio access network (RAN) functionalities, such as the O-RAN alliance [8]. O-RAN aims to reduce the cost of network deployment by using low-cost and white-box hardware for radio components. In the O-RAN alliance specifications regarding the AI/ML workflow in the radio access network, two main cognitive entities are designed to have a significant role in the management and lifecycle of AI/ML models. Specifically, the non-real-time (RT) RAN Intelligent Controller (RIC) and the near-RT RIC determine the optimization rationale of the O-RAN deployments and supervise the training, deployment and update of AI/ML models, depending on the corresponding time-scales of interest (e.g., the near-RT RIC operates in the range of 10-500 ms, while the non-RT RIC at a range greater than 500 ms).

However, the effective training of ML models requires access to a vast amount of data, directly collected from realistic 5G networks' infrastructure. The access to those data may be quite limited, either due to privacy issues or insufficient network deployment, especially in the last two years of the pandemic crisis. Therefore, the need for an artificial source of data that can substitute real-world measurements is crucial. In this framework, various research efforts have focused on the design and development of efficient 5G network emulators over the last years. In [9], the authors introduce the 5G K-SimNet, which provides various features such as dual-radio protocol, traffic management on multi-connectivity, and software-defined NFV. In [10], the authors have evaluated the performance of Simul5G, a 5G emulator which introduces lightweight models of UEs and gNBs, facilitating the simulation of large-scale 5G multicellular networks. In [11], the SyntheticNET simulator is introduced, based on python, incorporating the latest 3GPP 5G release as well as AI-based ray-tracing propagation modelling. The developed simulator can include additional AI/ML-based solutions for autonomous configuration and optimization of network parameters. In [12], an emulation platform for SDN-enabled 5G integrated fiber-wireless networks is presented and evaluated, which provides a transparent view of the 5G infrastructure to any SDN-based control plane. Nonetheless, the accurate performance evaluation of 5G networks that can incorporate real-time ML algorithms for KPI optimization, poses additional degrees of complexity in the designed solutions. Finally, a survey on 5G simulators is provided in [13].

This paper aims to demonstrate the support of intelligence control loop in O-RAN, by adopting open software solutions and embracing the concept of Zero-Touch Automation (ZTA). The developed open software solutions presented herein are thus compatible with the O-RAN specifications, as they are provided by AI/ML official guidelines. In addition, this work presents the implementation of a Deep Reinforcement Learning (DRL) algorithm in the radio access part. The main contributions of this work include: (i) the development and testing of a general-purpose RAN parameter generator that provides simulated measurements of 5G-compliant urban cells, following 3GPP guidelines and supporting UE mobility (https://github.com/FanisKar); (ii) the ZTA in the configuration of RAN components through the interlinking between the near-RT RIC and the RAN collected measurements; (iii) the testing and validation of O-RAN compliant intelligent functions (namely xApps) in a commercial, real-network near-RT RIC and associated inner modules, such as Databus, listeners, publishers and external containers; (iv) the optimization of the RAN experienced downlink throughput through power control, based on AI/ML and DRL. Finally, it is worth mentioning that all the presented material has been deployed in a virtualized server of a real testbed.

II. Architecture And Methodology

A. System Model and Parameters

An urban 5G network area is considered, consisting of K macro and/or micro-cells, which may have overlapping coverage areas, following the specifications of UMa/UMi cells detailed in [14]. Each cell contains the respective Radio Unit (RU) transmitter k (Tx $k, \forall k = 1, 2, \dots, K$), having a total bandwidth B. The selected 5G numerology dictates the number N of available Physical Resource Blocks (PRB) that will be used for physical-layer transmission ($n = 1, 2, \dots, N$). In addition, depending on the operational 5G frequency band, data are transmitted using the OFDM modulation scheme through an effective bandwidth W of each PRB. The UEs ($u = 1, 2, \dots, U$) that are located inside the network area can establish a link with a single PRB n of a particular RU k. The association status of each UE is expressed by the binary-valued association matrix $A_{k,n,u}$, which takes the value of 1 if a link is established between UE u and PRB n of RU k (or 0 otherwise). The sum-power constraint over all transmitting PRBs can be expressed by:

$$\sum_{n=1}^{N} P_{n,k} < P^{max}, \tag{1}$$

where the transmit power of RU k over PRB n is denoted by $P_{n,k}$, while P^{max} is the maximum allows transmission power per RU. A minimum power constrained $P_{n,k} > P^{min}$ is also assumed for each PRB band in order to take into account the idle/beaconing operational modes of the RU. At a specific time instance, the UEs that are located inside the 5G network area evaluate the beacon signals from all RU transmitters and are associated with the particular PRB of a specific RU that provides the maximum throughput. The Signal to Interference plus Noise Ratio (SINR) can be expressed as:

$$SINR_u^{n,k} = \frac{P_{n,k} \cdot G_{k,u}}{\left(\sum_{k' \neq k}^{K} P_{n,k'} \cdot G_{k',u}\right) + N_0} \tag{2}$$

where $G_{k,u}$ indicates the channel gain between UE u and RU k and N_0 is the noise power density at the UE device. The theoretical experienced throughput of UE u can be described by the Shannon formula:

$$R_u^{n,k} = W \cdot log_2\left(1 + SINR_u^{n,k}\right). \tag{3}$$

Hence, at each time instance, the received throughput by all possible RUs and PRBs are calculated for each UE and the links providing the maximum throughput are selected as association connections. Then, each UE has a probability F to move towards a random direction with a maximum velocity V_u and the same calculations for association are repeated through time.

Finally, the described RAN is controlled by a centralized intelligent entity (near-RT RIC) that is capable of regulating the power levels of the PRBs in all RUs at each time step. The main target of the cognitive controller is the maximization of the network-wide throughput ($\sum_{u=1}^{U} R_u^{n,k}$), thus significantly enhancing the QoS experienced by each UE inside the wireless network area. In the following subsection, we will present the individual building blocks to support both the 5G network simulator and the throughput optimization in the form of xApp developed in the centralized near-RT RIC.

B. Supporting xApps: Near-RT RIC Architecture

The internal structure of an xApp is shown in Fig. 1, where the individual boxes are essential for the development and implementation of the xApp core, xApp Database and the near-RT RIC Databus. The xApp API is used to provide communication with the near-RT RIC, while also enabling lifecycle and configuration management of the xApp. The xApp Database stores the xApp configuration parameters provided by the API and is responsible to deliver the configuration file inside the xApp core.

The internal structure of the xApp core includes the following modules: (i) *Database listener*: this inner module communicates with the xApp Database, retrieves the configuration and forwards it to the *xApp configuration*, where it is stored locally in order to be used in real-time processing, (ii) *Databus listener*: this module communicates with the *near-RT RIC Databus* and retrieves real-time RAN information, (iii) *In queue*: the data from the *Databus listener* are then provided to the *in queue* module and they can be shared across the xApp core, (iv) *Processor*: this module is the main building block of the xApp core where the data from *in queue* can be retrieved and processed according to the functional logic that the xApp is designed to perform (e.g., 5G network simulator, AI/ML model inference), (v) *Out queue*: the output data from the *processor* are placed in the *out queue* module. This will automatically trigger the *Databus publisher*, which in turn will publish the data in the *near-RT RIC Databus* and (vi) *Action taker*: this module can be optionally accessed from the *processor* and is responsible for implementing the output results (e.g., adjust RAN parameters).

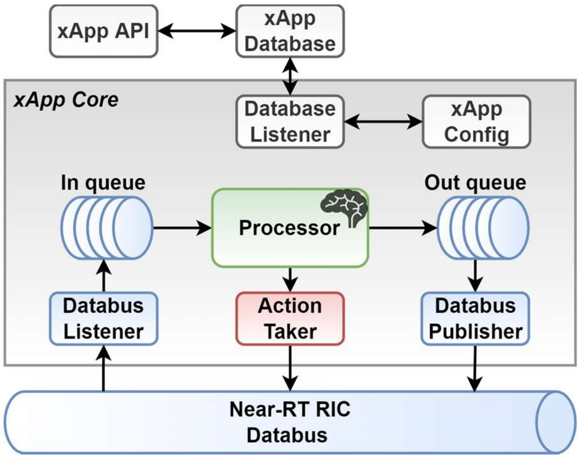


Fig. 1. Part of the near-RT RIC architecture supporting xApp delivery. The intelligent module of the xApp core is the Processor, whereas the other modules are dedicated for information exchange with the databus and database.

It is evident that the flexible architecture of the inner near-RT RIC modules enables the dynamic configuration of an xApp, allowing for efficient AI/ML model delivery and RAN parameter control.

C. POSS: a Python Open-Source Simulator for 5G systems

In the framework of this paper, a general-purpose flexible Python Open-Source Simulator (POSS) for 5G network systems is built in the form of xApp to provide physical-layer O-RAN measurements for further processing (see Fig. 2). The following input parameters can be adjusted for the configuration of the POSS: (i) Number of UMa and UMi cells that are considered in the network area and the location of the transmitters (RU topology), (ii) Maximum power budget limitation per RU and minimum power level of each PRB, (iii) Operating 5G frequency band, 5G numerology (spectrum channelization) and total available bandwidth that is used for downlink transmissions, (iv) Initial number of UEs inside the network area and their maximum velocity of the random walk model and (v) Probability of new UE entering inside the network coverage area from the cell borders in a given time slot.

The algorithm starts by randomly positioning the initial number of UEs inside the network area and randomly initializing the power level of each PRB of all RUs, while also respecting the maximum and minimum power budget limitations, as in (1). The channel gain $G_{k,u}$ for all UEs and all possible association links is calculated based on the relative distance between the UEs and each individual RU (5G specifications of UMa/UMi cells [14]). The key functionality of the simulator includes the interference calculations for each UE by also considering the accumulated interference caused by the non-serving RUs. After the calculation of the transmission power of PRB/RU and the channel gain, our algorithm calculates the SINR and the throughput that each UE can experience from all possible RU/PRB combinations, using (2) and (3). An association link is then formed among each UE and the PRB of the RU that provides the maximum QoS in terms of throughput.

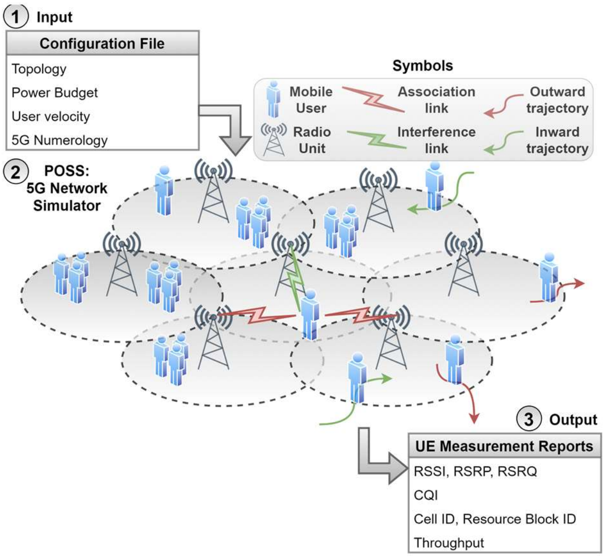


Fig. 2. Configuration of POSS including the input parameters, the provided internal functionalities (association, pathloss, channel model, interference, mobility, handover) and the output UE measurement reports.

At each time instance, seven metrics can be produced by the POSS once the association links are established, namely the (Receive Strength Signal Indicator (RSSI), Reference Signal Received Power (RSRP), Reference Signal Received Quality (RSRQ), Channel Quality Indicator (CQI), associated RU ID, associated PRB ID and UE experienced throughput.

Before moving to the next time slot, each UE follows a random-walk mobility model, whereas the border users have a probability to exit the network area. In addition, new users may enter the border cell areas with a probability specified in the configuration input file. Hence, the POSS is able to support time-varying mobile users, while also handling potential handovers. Output user-specific time series may be then exploited for further analysis, such as AI/ML model training and inference, network statistics or analytics.

D. Throughput Optimization algorithm

Here we consider the problem of network-wide throughput maximization through power regulation. In general, improvements in the network throughput can be achieved by increasing the power level of the associated links, while simultaneously mitigating the co-channel interferences. Thus, the algorithm exploits the Shannon's formula to calculate the sum of individual data rates and, based on the measurement reports in each cell, aims to maximize the total network-wide throughput. Formally, the algorithm objective, according to (3), can be expressed as:

$$\max_{p\epsilon P} \sum_{k\epsilon K} \sum_{n\epsilon N} W \cdot log_2\left(1 + SINR_u^{n,k}\right), \qquad (4)$$

where P, K, N are the sets of power configuration, RUs, PRBs, respectively, while the constraints are discussed in Section II.A.

In DRL terminology, the centralized *agent* (the near-RT RIC) interacting with a telecom *environment* (cellular areas) aims to maximize the *objective function* (the network throughput) by observing the *state space* (the measurement reports) and taking *actions* (power level adjustments) [15]. Specifically, the agent-environment interaction cycle involves the following procedure: given a state observation (i.e., CQI reports for each RU/PRB pair), the agent performs an action (selection of a PRB from each RU and increase, decrease or maintain its power level by a predefined power step) and receives a reward (difference between the current and previous total network). Note that in the case of a non-occupied PRB, the respective state element is set to -1. Following a trial-and-error approach, the agent gradually converges in power control policy that ensures increased network utility.

E. Network Automation through xApps

In this section, the practical implementation and interaction of the POSS and the DRL agent, which can be deployed in the form of xApps in the near-RT RIC, is presented (Fig. 3). To this end, 5G-compliant measurement reports of UEs in a configured network topology are generated by the POSS simulator (xApp1) and acknowledged to the near-RT RIC Databus (Step 1). Consequently, the pre-trained DRL agent (xApp2) reads the measurement report from the Databus (Step 2) and, after combining all the RU-specific reports, uses them as input to infer the DRL model and provide corrective power levels to ensure throughput increments. Finally, the RU/PRB power suggestions are published back to the Databus (Step 3) and applied to the next network instance (Step 4). This process is continuously repeated to visualize the time-course of the model inference performance. Note that in the beginning of each interaction cycle the environment dynamically changes not only due to the power corrections, but also due to the mobility of the UEs (see functionalities of the POSS in section II.C).

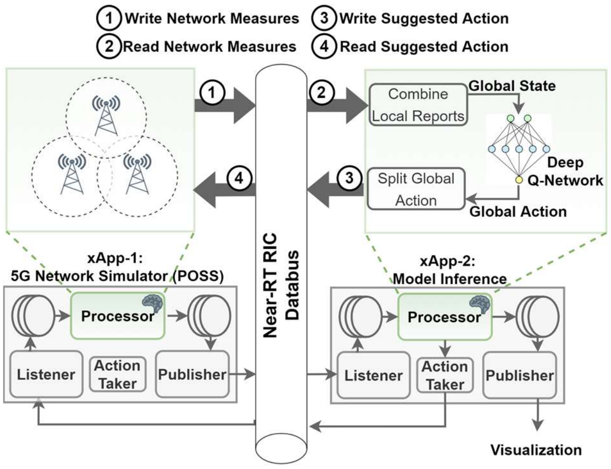


Fig. 3. Development of xApps for closed-loop automation: xApp1 (POSS for measurement report generation) and xApp2 (DRL optimization agent), interacting through the near-RT RIC Databus. Green-shaded boxes indicate the internal functional logic of xApps.

It is worth mentioning that, during the real network operation, the measurement report generated by the POSS simulator is actually the real network measurements, thereby can be directly provided to xApp2 for throughput maximization. Additionally, several xApps can be deployed in the near-RT RIC that either contain different ML models for the same optimization objective (throughput maximization) or even different optimization objectives concerning the RAN operation (for instance bandwidth allocation, energy efficiency maximization, etc.). In this context, diverse objective-specific AI/ML models can be deployed, dockerized and stored in a dedicated ML catalogue, being available for potential future inference. Depending on the real-time policy provided by the higher layers (e.g. the non-RT RIC in the Service and Management Orchestration layer) and/or the Intent-based manager, the near-RT RIC may select from the ML catalogue the appropriate model that aligns with the targeted optimization intent.

F. Dockerization and Deployment of xApps

Given that the implementation of the xApps framework presented above was deployed in a commercial near-RT RIC product, in this section we describe some of the key development aspects. To achieve co-operation between the involved development modules, we used the Kubernetes cluster framework. The xApp1 is a wrapper application that uses the POSS functions, providing a Redis and an Apache Kafka interface to store or read data. In every iteration, the xApp1 is responsible for getting the output data of the POSS and writing them inside a Kafka message broker (near-RT RIC Databus) using its implemented interface. For that purpose, the main function of our xApp1 application reads information on how to access the Kafka bus and Redis from a configuration file. The xApp1 is dockerized and can be easily deployed on a docker container or a Kubernetes cluster with the help of a Dockerfile that is written inside the code-base. Running on a cluster will also ensure maximum availability for our application.

The xApp2 wraps our inference logic. This means that, at a given timeslot of the network operation, the xApp2 performs the DRL model inference, upon reading the data written by xApp1 on the Kafka bus. The inference result (i.e., the suggested power vector) is then written back to the Kafka bus on another topic. It should be noted that when a different model is required for usage, a separate xApp is required to be deployed. Since multiple xApps deployments are required to infer multiple models (e.g. several models that are simultaneously running in the near-RT RIC), a mechanism to support automated model serving would provide generalized model delivery. To support such flexibility and avoid the deployment of numerous xApps, automated model serving tools (e.g. Tensorflow serving) could be alternatively incorporated. For instance, if Tensorflow serving has been included in the xApp2 logic, the inference operation can be effortlessly performed via an HTTP Rest API call.

III. Simulation Results

In this section, we firstly present simulation results to highlight the functionalities of the POSS. Then, results concerning the hyper-parameter tuning during the training phase of the DRL agent are demonstrated. Finally, subsection III.C illustrates the interaction and simultaneous operation of the two xApps that are deployed in the near-RT RIC. The presented results for the POSS and the DRL agent were conducted in Python 3.8 using the TensorFlow (version 2.3), Keras and Scikit-Learn libraries. The hardware used to run the scripts was a personal PC (CPU i7-2600; 3.4 GHz; RAM 12 GB; no GPU usage). We considered the operating frequency of 6 GHz (bandwidth of B = 20 MHz) and omnidirectional antenna radiation patterns.

A. xApp1: Simulator Functionalities

In this subsection, the functional logic and the different configuration capabilities of the developed 5G simulator (POSS) are demonstrated. Indicatively, three configuration scenarios of the simulator are considered, parameterized as: (i) a full-capacity scenario with a 3-cell topology and numerology 5 (3 PRBs per RU, each one with bandwidth 5.76 MHz) (ii) a full-capacity scenario with a 4-cell topology and numerology 4 (6 PRBs per RU, each one with bandwidth 2.88 MHz) (iii) a full-capacity scenario with a 5-cell topology and numerology 3 (12 PRBs per RU, each one with bandwidth 1.44 MHz).

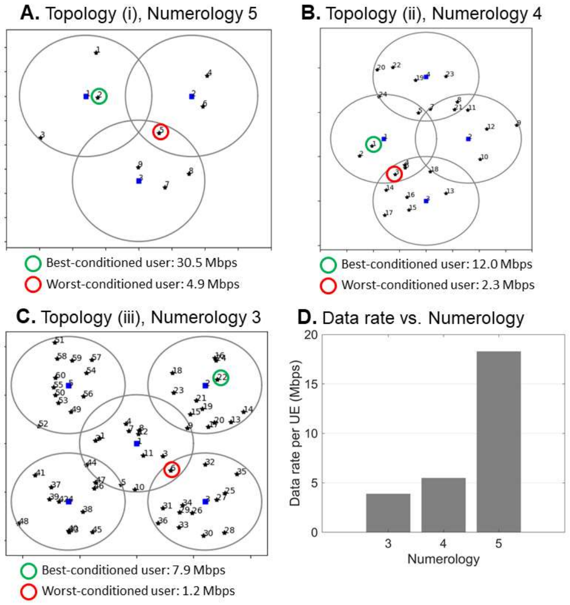


Fig. 4. Configuration capabilities of POSS. A – C. Network topologies consisting of different number of cells and different 5G numerology. D. Average UE data-rate relative to the employed 5G numerology.

In each scenario, diverse user mobility speeds can be established to represent heterogeneous network demand points. Typical categories include fixed reception points/PCs (0 m/s), pedestrians (1m/s) and vehicles (10 – 20 m/s). Figs. 4a – 4c depict a snapshot of the three considered topologies, along with the instantaneous allocated data rate experienced by 2 extreme case users; namely 1 best-conditioned user (nearby its associated cell) and 1 worst-conditioned user (located in the overlap area of the server and interferers). Firstly, it is evident that the selected numerology and channel segmentation strongly affects the user-specific allocated throughput. This is illustrated in Fig. 4d, where the average UE data rate is positively correlated with the numerology. Contradictorily, the UE capacity decreases with the numerology, as high-degree segmentation creates higher number of available channels per transmitter. As expected, it is also evident that the experienced data rate decreases for the border/overlap-area users (circled in Fig. 4a – c). Note that, the possible POSS configurations are not limited to those presented here.

B. xApp2: Training and Evaluating the DRL agents

As already mentioned, the training of the DRL agent for throughput maximization is done offline, and the pre-trained DRL agent is deployed in the near-RT RIC as an xApp. To this end, the training process was performed with simulated measurements of the POSS simulator and herein we present indicative results concerning the DRL hyper-parameter tuning for a network topology of 3 cells (5G numerology is 3, initial number of users is 10). In each training episode, before letting the agent start to interact with the environment, the initial power levels of all PRBs and RUs are set to the average power level. This means that the DRL algorithm begins with a good initial throughput status in the network, since the average power levels ensure a reasonable trade-off between the obtained links' throughput, while mitigating their harmful interference in the rest of the co-channel transmissions. Typical parameters to be fine-tuned include the learning rate, the discount factor and the hidden layers of the neural network [16], [17]. Noteworthy, the discount factor (γ) is used in the Bellman equation for updating the Q-values and is associated with the extent to which the agent prefers immediate (γ=0) or future (γ=1) rewards [18].

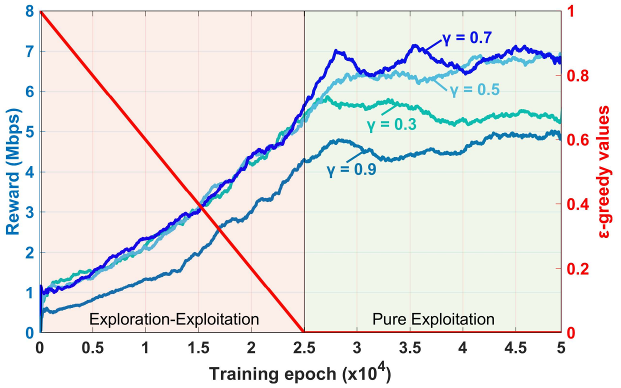
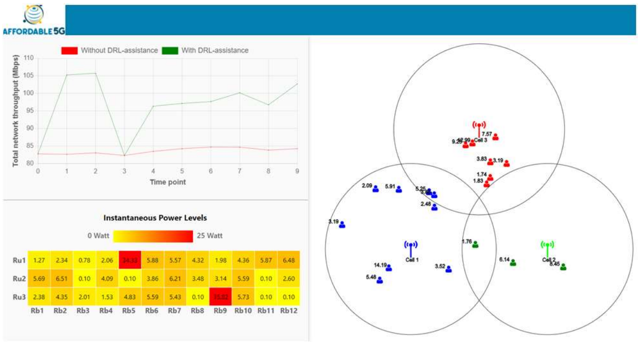


Fig. 5. Learning curves for different values of γ during the training phase of the DRL agent. The red curve corresponds to the ε-greedy selected policy.

As readily observed from Fig. 5, the agent begins by performing random actions on the wireless environment, i.e., selecting randomly the PRBs and increasing, decreasing or maintaining the power level (exploration phase). Gradually the agent advances to the exploitation phase, thus performing actions that, based on his experience, will give him the maximal rewards. It is evident from Fig. 5 that the optimal values of the discount factor are presented for median values (0.5 or 0.7), and not for extremely low or extremely high settings. This implies that the agent performs beneficially when balances between the old and the expected return (or between past values and future estimations of the reward).

C. Overall Assessment: Integrating xApp1 and xApp2

Here we demonstrate the integration of the two xApps to concretely visualize the achieved closed-loop automation. The right panel of Fig. 6 depicts a snapshot of the network, where the moving UEs located inside the topology are color-coded in line with their associated RU. We observe (in Mbps) the UE-specific throughput denoted above each user. In the left panel, the DRL-assisted power levels (in Watts) are provided as a heatmap, whereas the total throughput derived by the DRL suggestion is compared to the total data rate obtained by the average power level policy (initial state) during a period of 10 time instances. Benchmarking comparisons can be found in [16], [17].



	Rb1	Rb2	Rb3	Rb4	Rb5	Rb6	Rb7	Rb8	Rb9	Rb10	Rb11	Rb12
Ru1	1.27	2.34	0.78	2.06	24.33	5.88	5.57	4.32	1.98	4.36	5.87	6.48
Ru2	5.69	6.51	0.10	4.09	0.10	3.86	6.21	3.48	3.14	5.59	0.10	2.60
Ru3	2.38	4.35	2.01	1.53	4.83	5.59	5.43	0.10	15.82	5.73	0.10	0.10

Fig. 6. A snapshot of the network operation following the xApp1 and xApp2 integration. The total allocated throughput for 10 time slots is depicted (upper left) with and without DRL assistance, whereas the instantaneous power vectors are also shown (lower left) as heatmap.

Evidently, the DRL scheme is capable of enhancing the network-wide throughput, noticing at least equal performance to the average scheme. Since there is a possibility of UEs being located in the cross-cell areas (such as in time instance #3), the throughput increment may not be feasible in every time slot. Note that, given the mobility of UEs, the reception conditions and the UE pathloss are recalculated in each time slot, thereby the DRL inference is time-varying.

IV. Conclusion

This article presented the key aspects of the current technological advances underlying O-RAN deployments. After describing the internal structure of the RIC to support extended optimization applications (xApps), the implementation and validation of the xApps were demonstrated. Indicatively, we considered 2 xApps to enable closed-loop, automated and self-configured optimization for radio resource management. A general-purpose and open-source simulator (xApp1) was also provided for testing the performance of a throughput maximization targeted DRL agent (xApp2) in realistic measurements. Finally, quantitative simulations for training, testing and visualizing the AI/ML models were presented aiming to concretely illustrate the procedure for deploying and running xApps in the O-RAN architectures.

Acknowledgment

This work has been partially supported by the Affordable5G project, funded by the European Commission under Grant Agreement H2020-ICT-2020-1, number 957317 through the Horizon 2020 and 5G-PPP programs (www.affordable5g.eu/). The authors thank Simon Pryor (Accelleran) for his contribution to the near-RT RIC architecture.

References

[1] A. Kaloxylos, A. Gavras, D. Camps Mur, M. Ghoraishi and H. Hrasnica, “AI and ML—Enablers for Beyond 5G Networks”, *Zenodo*, 2020.

[2] P. Trakadas et al., “A cost-efficient 5G non-public network architectural approach: Key concepts and enablers, building blocks and potential use cases”, *Sensors*, vol.21, issue 16, 2021, pp. 5578.

[3] N. Nomikos et al., “A UAV-based moving 5G RAN for massive connectivity of mobile users and IoT devices”, *Vehicular Communications*, vol. 25, 2020, pp. 100250.

[4] P. Alemany et al., “A KPI-enabled NFV MANO architecture for network Slicing with QoS,” *IEEE Communications Magazine*, vol. 59, no. 7, pp. 44-50, July 2021.

[5] M. Peuster et al., “Introducing automated verification and validation for virtualized network functions and services”, *IEEE Communications Magazine*, vol. 57, issue 5, 2019, pp. 96-102.

[6] M. Filo, C. H. Foh, S. Vahid and R. Tafazolli, “Performance analysis of ultra-dense networks with regularly deployed base stations,” *IEEE Transactions on Wireless Communications*, vol. 19, no. 5, pp. 3530-3545, May 2020.

[7] ET Michailidis, “Three-dimensional modeling of mmWave doubly massive MIMO aerial fading channels”, *IEEE Transactions on Vehicular Technology*, vol. 69, issue 2, 2019, pp. 1190-1202.

[8] The O-RAN alliance, https://www.o-ran.org/

[9] S. Choi, S. Kang, S. Bahk, J. Song and T. T. Kwon, “5G K-SimNet: Network simulator for evaluating end-to-end performance of 5G cellular Systems,” *2018 IEEE International Symposium on Dynamic Spectrum Access Networks* (DySPAN), 2018, pp. 1-2, doi: 10.1109/DySPAN.2018.8610468.

[10] G. Nardini, G. Stea and A. Virdis, “Scalable real-time emulation of 5G networks with Simu5G,” *IEEE Access*, vol. 9, pp. 148504-148520, 2021.

[11] S. M. A. Zaidi, M. Manalastas, H. Farooq and A. Imran, “SyntheticNET: A 3GPP compliant simulator for AI enabled 5G and beyond,” *IEEE Access*, vol. 8, pp. 82938-82950, 2020.

[12] A. Kretsis, P. Soumplis, P. Kokkinos and E. A. Varvarigos, “An SDN emulation platform for converged fiber-wireless 5G networks,” *2021 International Conference on Computer Communications and Networks* (*ICCCN*), 2021, pp. 1-7.

[13] P. Gkonis, et al., “A comprehensive study on simulation techniques for 5G networks: state of the art results, analysis, and future challenges”, Electronics, vol. 9, issue 3, 2020, pp. 468.

[14] 3GPP Technical Document, “Study on Channel Model for Frequencies From 0.5 to 100 GHz,” 2017.

[15] A. Giannopoulos, S. Spantideas, N. Kapsalis, P. Karkazis, and P. Trakadas, “Deep Reinforcement Learning for Energy-Efficient Multi-Channel Transmissions in 5G Cognitive HetNets: Centralized, Decentralized and Transfer Learning Based Solutions,” *IEEE Access*, vol. 9, pp. 129358–129374, 2021.

[16] S. Spantideas, A. Giannopoulos, A. Kalafatelis, C. Capsalis, N. Kapsalis, and P. Trakadas, “Joint Energy-efficient and Throughputsufficient Transmissions in 5G Cells with Deep Q-Learning,” in *Proceedings of the 2021 IEEE International Mediterranean Conference on Communications and Networking (MeditCom)*, 2021, pp. 7–10.

[17] A. Giannopoulos, S. Spantideas, C. Tsinos, and P. Trakadas, “Power Control in 5G Heterogeneous Cells Considering User Demands Using Deep Reinforcement Learning,” in *IFIP Advances in Information and Communication Technology*, 2021, vol. 628, pp. 95–105.

[18] A. Giannopoulos, et al., “WIP: Demand-driven power allocation in wireless networks with deep Q-learning”, in *2021 IEEE 22nd International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, 2021, pp. 248-2