Policy-Gradient-Based Reinforcement Learning for Computing Resources Allocation in O-RAN

Mahdi Sharara, Turgay Pamuklu, Sahar Hoteit, Véronique Vèque, Melike Erol-Kantarci

Policy-Gradient-Based Reinforcement Learning for Computing Resources Allocation in O-RAN

Mahdi Sharara*, Turgay Pamuklu†, Sahar Hoteit*, Véronique Vèque*
and Melike Erol-Kantarci†
*Laboratoire des Signaux et Systèmes, Université Paris Saclay-CNRS-CentraleSupélec, France
†School of Electrical Engineering and Computer Science, University of Ottawa, Canada
Emails: {mahdi.sharara, sahar.hoteit, veronique.veque}@universite-paris-saclay.fr,
{turgay.pamuklu, melike.erolkantarci}@uottawa.ca

***Abstract*—Open Radio Access Network (O-RAN) is a novel architecture aiming to disaggregate the network components to reduce capital and operational costs and open the interfaces to ensure interoperability. In this work, we consider the problem of allocating computing resources to process the data of enhanced Mobile BroadBand (eMBB) users and Ultra-Reliable Low-Latency (URLLC) Users. Supposing the processing of users' frames from different base stations is done in a shared O-Cloud, we model the computing resources allocation problem as an Integer Linear Programming (ILP) problem that aims at fairly allocating computing resources to eMBB and URLLC users and optimizing the QoS of URLLC users without neglecting eMBB users. Due to the high complexity of solving an ILP problem, we model the problem using Reinforcement Learning (RL). Our results demonstrate the ability of our RL-based solution to perform close to the ILP solver while having much lower computational complexity. For a different number of Open Radio Units (O-RUs), the objective value of the RL agent does not deviate from the ILP objective by more than 6%.**

***Index Terms*—O-RAN, Integer Linear Programming, Reinforcement Learning, Computing Resources Allocation**

I. Introduction

The demand for data is massively growing every year. In 2030, the global mobile traffic is expected to reach 5016 EB/month [1]. To respond to the growing demand, cellular network architecture continues to evolve. Compared to previous technologies, cellular networks are increasingly moving towards a virtualized, softwarized, and open architecture. Recently, a novel paradigm, Open Radio Access Network (O-RAN), has emerged [2].

O-RAN is a novel architecture that disaggregates the radio access network; some functions are executed on Distributed Units (DUs), while upper layers functions are executed on Centralized Units (CUs). Both units could be softwarized and could run as virtual machines. Through standardizing open interfaces between different network components, O-RAN paves the way for multi-vendor architecture where various interoperable network components can belong to different vendors [2]. This would encourage competition and make operators less dependent on a limited number of telecom vendors. Virtualization and multi-vendor architecture would also help operators reduce their CAPEX and OPEX [3].

A main pillar of O-RAN is the incorporation and standardization of machine learning (ML) and data-driven algorithm. With the availability of enormous amounts of data, it is possible to dynamically adapt the network parameters to meet the instant requirements. In other words, the network could use ML, especially Reinforcement Learning (RL), to autonomously learn optimal policies by interacting with the environment and make decisions accordingly. Such decisions could involve real-time decisions such as Resource Block (RB) scheduling, power allocation, or non-real-time decisions. Given the high dimensionality of the network, ML models are low-complexity candidates that have the potential to solve different problems in mobile networks. O-RAN defines RAN Intelligent Controllers (RICs), allowing operators to deploy custom control loops and use AI-enabled algorithms [2].

In 5G and beyond, different services are expected to coexist. For example, enhanced Mobile BroadBand (eMBB) and Ultra Reliable Low Latency (URLLC) services have different requirements; the former aims at providing high data rates, while the latter aims at transmitting data with minimal delay and low packet loss rates. For example, URLLC services could be needed to realize vehicular communications for autonomous vehicles [4] or to realize factory automation [5]. URLLC and eMBB services could be isolated logically while running on the same physical radio and computational hardware. Such a mechanism is known as Network Slicing (NS) [1].

Operators would try to provision slices with the enough required resources; however, the provisioned resources could become insufficient because of non-optimal provisioning or under-provisioning to cut CAPEX and OPEX, network dynamicity, evolving demands, or network failures. During the transitory time before sufficient resources could be re-provisioned to slices, the Quality of Service (QoS) could no more be guaranteed. Still, it remains necessary for the operator to optimize the performance (i.e., throughput, fairness) given the resource shortage. We are considering the case where losing URLLC services will not cause fatal incidents but would affect the application's performance; otherwise, it will be mandatory for the operator to completely avoid having a shortage of resources. On the other hand, the operator needs to balance the allocation of resources to users of both eMBB and URLLC services and can not neglect a service entirely.

This paper considers the problem in which the eMBB and URLLC services compete for limited and insufficient

computing resources. In such a scenario, it will not be possible to process the frames of all users in the required amount of time, and non-processed frames will be lost. The operator will have to select frames for processing and neglect others due to the lack of resources while maximizing fairness among users from the different services. We model the problem as an Integer Linear Programming (ILP) problem. However, given the high complexity of solving the NP-Hard ILP problem, we opt for using a policy gradient-based RL algorithm that should perform as close as possible to the ILP solver. Hence, we model the problem as a Markov Decision Process (MDP) and present an RL algorithm to solve it. Then, we analyze the performance of the ILP problem and the RL algorithm in addition to two well-known low-complexity heuristics, Round-Robin (RR) and Proportional Fairness (PF) [6].

The rest of the paper is organized as follows: Section II presents the state-of-the-art. The context and problem formulation is presented in section II. Then, the RL model and algorithm are presented in III. The simulation environment and results are shown in V, and finally, the work is concluded in section VI.

II. Related Work

O-RAN has received massive attention from both academia and industry recently. The paper in [7] presents a comprehensive survey on O-RAN, its architecture, open interfaces, and the research challenges facing O-RAN. In [2], the authors demonstrate the feasibility of near-real-time radio access control using deep reinforcement learning. They test the RL-based scheduling algorithms on a large-scale O-RAN-compliant softwarized cellular network called Colosseum. Using Colosseum, the authors of [8] develop an orchestration model in O-RAN, called OrchestRAN. The model is required to determine the optimal set of data-driven algorithms, avoiding conflicts between them. The model is general and has to select the appropriate ML model for each task in the network. Additionally, the model decides the optimal place of network components (e.g., CU, DU, RIC), considering that multiple network nodes could share models, but the Quality of Service should be respected. For example, if a CU manages two DUs, and the two DUs use a common ML model, it could be possible to implement the model in the CU and transmit the decisions to both DUs. However, it is possible that some delay constraints would force OrchestRAN to implement the model on each of the DUs; thus, extra computing resources would be used to respect QoS requirements.

Various studies demonstrate the ability of RL-based solutions to replace traditional network functions. In [9], an RL model has been used aiming at satisfying the demands of users from different slices/services. These demands include communication and computational resources requests. In the context of network slicing, [10] proposes two RL algorithms to realize upper and lower control. In particular, a Deep Deterministic Policy Gradient (DDPG) is used for the lower control: Resource Blocks (RBs) allocation and power allocation. On the upper level, a Double Deep Q-Network-based RL is used to learn the optimal RAN slicing strategies. Authors of [11] used Policy Gradient RL to learn the optimal functional split to minimize the computational and routing cost. To satisfy the latency requirements of URLLC users without hindering the eMBB throughput, [12] proposes a Q-learning algorithm responsible for resource and power allocation. The proposed algorithm manages to improve latency and reliability with respect to baseline algorithms.

In our previous works [13] [14], we have considered computing resource allocation in Cloud-RAN architecture. Considering the problem of limited computing resources where frames compete for processing resources, [13] introduces Integer Linear Programming models that maximize throughput and fairness allocation. The models permit coordinating between the radio and computing schedulers such that the radio parameter Modulation and Coding Scheme (MCS) index can be modified at the request of the computing scheduler. For a given frame, the MCS index could be decreased and would lead to reduced throughput, but the computing scheduler would be able to allocate the required computing resources to process this frame. Knowing that solving an ILP problem is NP-Hard, [14] presents a Recurrent Neural Network Model that aims at performing as close as possible to the ILP model presented in [13] but with lower complexity.

In contrast with the state-of-the-art, our work in this paper considers the computing resource allocation problem in a multi-service (i.e., eMBB and URLLC) O-RAN environment. We model the problem as an ILP problem aiming at maximizing the minimum transmission opportunities per user while maximizing the number of processed frames. Then we propose a policy-gradient-based RL algorithm to solve the problem suboptimally.

III. Context and Problem Formulation

A. O-RAN Context

This paper addresses an O-RAN multi-vendor-based slicing scenario [15]. Figure 1 illustrates the details of that scenario. In this network, vendor A supplies the O-RUs to the operator. The radio resources are shared with a predetermined ratio by two slices providing eMBB and URLLC services, respectively. The services have different delay requirements; URLLC has a tighter deadline and a smaller frame size in comparison with eMBB.

O-RAN follows CU/ DU split F1 between the vO-CU and vO-DU, while it follows functional split Option-7.2 between O-RU and O-DU [16]. We assume that a single dedicated O-DU manages each slice of an O-RU. The O-DU and O-CU are software-based components. Hence, they are virtual O-DU (vO-DU) and virtual O-CU (vO-CU). Furthermore, vO-DUs and vO-CUs of the URLLC slice are administered in the same edge O-Cloud due to the strict delay requirements of these services. On the other hand, the vO-CU of the eMBB service is deployed in a regional O-Cloud, which has a lower leasing cost, while its vO-DU is deployed in the Edge Cloud. Different vendors provide the virtual network components of each slice;

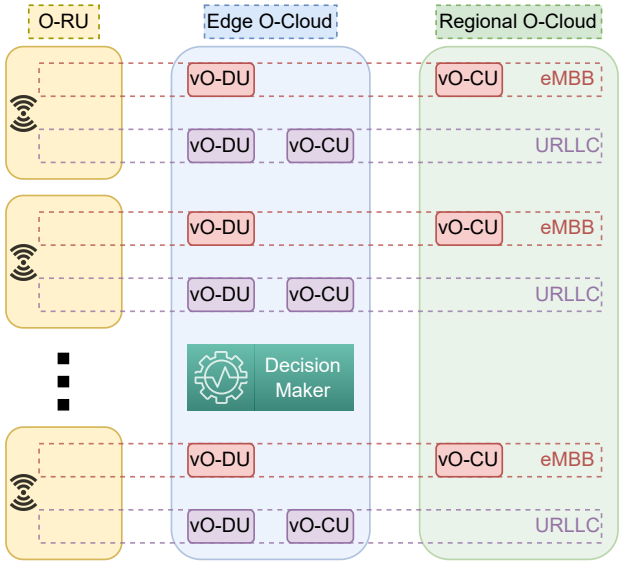


Fig. 1: Multi-vendor-based slicing scenario. Vendor A provides the O-RUs of the considered network. Vendors B and C provides the components of eMBB and URLLC slices, respectively.

the eMBB slice is supplied by vendor B, and the URLLC slice is supplied by vendor C.

O-RAN relies on near-RT-RIC and non-real-RT-RIC to train and execute AI and ML algorithms. Each of these RIC support control loops with a time scale larger than 10ms. To respect smaller time scale requirements (i.e., computing scheduling is done every 1ms), we propose an independent decision maker that improves the quality of service, has control over the CUs and DUs, and runs in the edge O-Cloud [17]. Knowing that the high physical layer functions such as decoding are executed in the vO-DUs, we suppose that all these vO-DUs share the same computing resources and that this intelligent decision maker controls the allocation of computing resources to these vO-DUs. The idea of implementing and standardizing intelligence for real-time decision-making that runs on the CU or the DU is still in the research phase, and it is not yet standardized [18].

Considering the uplink direction in which the resource heavy-decoding function is executed, the traffic may unpredictably surge on some occasions, making it impossible to provide the O-DUs with the required resources immediately. Hence these O-DUs could fail to execute the decoding functions for all users. Given that there is a deadline for execution equal to 2 ms due to HARQ [13], some of the O-DUs will fail to process some of their users' frames, and this would trigger the HARQ mechanism; the frames will be retransmitted, and radio resources will be reserved for the future retransmission. We aim to use RL to learn which users should be admitted and which should halt their transmission. This would allow users whose data will not be processed to save their transmission power.

We consider a set of O-RUs $\mathcal{R}$. For each O-RU r, there exists a set of eMBB users $\mathcal{U}_r^E$ and a set of URLLC users $\mathcal{U}_r^U$. The combined sets of eMBB and URLLC users is $\mathcal{U}_r$. The total number of users from all O-RUs is N. Fairness allocation of resources among users should be optimized over multiple Transmission Time Intervals (TTIs). Hence, we suppose each set of users persists in the network for T consecutive TTIs, and our goal is to apply the optimization problem over a set $\mathcal{T}$ of TTIs, where $T = |\mathcal{T}|$. The deadline to process eMBB users is d_E, and it is d_U for URLLC users. The set of CPUs available to process users' frames arriving in the same TTI is $\mathcal{C}$. We suppose that each user maintains the same number of RBs and the MCS index during all the TTIs. Hence the throughput of a user for a given MCS and number of RBs at TTI t is $b^{r,u}$ and it takes $e_t^{r,u}$ amount of time to process a user frame at the O-Cloud. Both the throughput and processing time depend on the MCS and the number of RBs. We recall that the Modulation and Coding Scheme index (MCS) defines the modulation and the code rate used to transmit a frame. The number of RBs and the MCS index together determine the Transport Block Size (TBS), which is the number of bits transmitted in one Transmission Time Interval (TTI) [19]. Additionally, the processing time depends on these two parameters [20]. As our goal is to ensure a fair allocation of computing resources among users, we evaluate the fairness using Jain's Fairness index [21] defined as follows:

$$J_I = \frac{\left(\sum_{r\in\mathcal{R}}\sum_{u\in\mathcal{U}_r}\sum_{t\in\mathcal{T}}\frac{x_{t,c}^{r,u}}{T}\right)^2}{\left(N\times\sum_{r\in\mathcal{R}}\sum_{u\in\mathcal{U}_r}\sum_{t\in\mathcal{T}}\left(\frac{x_{t,c}^{r,u}}{T}\right)^2\right)}$$

where $\sum_{t\in\mathcal{T}}\frac{x_{t,c}^{r,u}}{T}$ defines the ratio of the allocation over the demand for user $u \in \mathcal{U}_r$. We note that Jain's fairness index is maximal (equal to 1) when all users have the same ratio of allocation over the demand.

B. ILP Model

The Integer Linear Programming (ILP) model we propose for computing resources allocation to eMBB and URLLC users is defined as follows:

$$\text{maximize} \quad \lambda(a_E + \sum_{r\in\mathcal{R}}\sum_{u\in\mathcal{U}_r^E}\sum_{t\in\mathcal{T}}\sum_{c\in\mathcal{C}}\frac{x_{t,c}^{r,u}}{T\times N})+$$
$$(1-\lambda)(a_U + \sum_{r\in\mathcal{R}}\sum_{u\in\mathcal{U}_r^U}\sum_{t\in\mathcal{T}}\sum_{c\in\mathcal{C}}\frac{x_{t,c}^{r,u}}{T\times N}) \quad (1)$$

$$\text{subject to} \quad x_{t,c}^{r,u} \in \{0,1\},\ \forall r\in\mathcal{R}, u\in\mathcal{U}_r, t\in\mathcal{T}, c\in\mathcal{C} \quad (2)$$

$$\sum_{c\in\mathcal{C}} x_{t,c}^{r,u} \le 1,\ \forall r\in\mathcal{R}, u\in\mathcal{U}_r, t\in\mathcal{T} \quad (3)$$

$$\sum_{r\in\mathcal{R}}\sum_{t\in\mathcal{T}}\left(\sum_{u\in\mathcal{U}_r^E} x_{t,c}^{r,u}e_t^{r,u} + \sum_{u\in\mathcal{U}_r^U} x_{t,c}^{r,u}e_t^{r,u}\right) \le d_E,$$
$$\forall c\in\mathcal{C} \quad (4)$$

$$\sum_{r\in\mathcal{R}}\sum_{t\in\mathcal{T}}\sum_{u\in\mathcal{U}_r^U} x_{t,c}^{r,u}e_t^{r,u} \le d_U, \forall c\in\mathcal{C} \quad (5)$$

$$\frac{\sum_{t\in\mathcal{T}}\sum_{c\in\mathcal{C}} x_{t,c}^{r,u}}{T} \ge a_E, \forall r\in\mathcal{R}, u\in\mathcal{U}_r^E \quad (6)$$

$$\frac{\sum_{t\in\mathcal{T}}\sum_{c\in\mathcal{C}} x_{t,c}^{r,u}}{T} \geq a_U, \forall r \in \mathcal{R}, u \in \mathcal{U}_r^U \quad (7)$$

The objective function (1) aims to maximize the minimum number of transmissions a user makes. a_E anda_U are auxiliary decision variables that set the minimum ratio of granted throughput to demanded throughput for eMBB users and URLLC users, respectively. As it is more rational to use all the available resources than to keep the CPU idle, even if some users will get more chances than others, the objective function includes the average number of transmissions that users have made. So a minimum number of transmissions for all users will be guaranteed, and then the remaining resources will be given to some users. Moreover, the objective separates the URLLC users from eMBB users, allowing the operator to control the prioritization level using the parameter λ. It is a value between 0 and 1, and it is up to the operator to select its priorities. The constraint in (2) defines $x_{t,c}^{r,u}$ as a binary integer variable. This variable is equal to 1 if and only if a user $u \in \mathcal{U}_r$ transmits at TTI t and is processed at CPU core c. Constraint (3) ensures that a user's unique transmission is not processed more than once. Constraint (4) ensures that the eMBB deadline is respected. Constraint (5) guarantees that the processing deadline of admitted URLLC users is respected. Finally, constraint (6) and (7) set the minimum of number of transmissions that eMBB and URLLC users get using the auxiliary variables a_E and a_U.

According to [22], solving an ILP problem is an NP-hard; hence we try to solve the problem using RL.

IV. Reinforcement Learning Model

Consider an episodic RL model. Each TTI is an episode that consists of multiple steps, where at each step, a user is selected for transmission and processing. The Markov Decision Process (MDP) is:

A. State

At TTI t during an step i, one user will be selected. Let $b^{r,u}$ be the throughput of user $u \in U_r$ that uses MCS $m_t^{r,u}$ over $n_t^{r,u}$ number of RBs. Let $e_t^{r,u}$ be the required execution time of the user's frame. Let μ_t^i be the available computing resources at step i at TTI t, $service^{r,u}$ indicates the service type: eMBB or URLLC, and $h_t^{r,u}$ is the total number of a user's previous transmissions before TTI t. Then:

$$\delta_{r,u}^{i,t} = \{e_t^{r,u}, b_t^{r,u}, h_t^{r,u}, service^{r,u}\}$$

The state at step i of TTI t is defined as

$$s_t^i = \Big\{\delta_{r,u}^{i,t} : \forall r \in \mathcal{R}, u \in \mathcal{U}_r, a_t^j \neq u, \forall j < i, \\ e_t^{r,u} \leq \mu_t^i\Big\}$$

The state space $\mathcal{S}$ at the initial step i is of dimensions: $\prod_{r\in\mathcal{R}}\prod_{u\in\mathcal{U}_r}\mathbb{R}^2\times\mathbb{N}\times 2$. The state space dimensions decrease after making each selection. After each step, a selected user should not be reselected; it should be removed from the action space. Suppose that user $u \in \mathcal{U}_r$ has been selected. Given our adopted Neural Network architecture, as shown in the next part, removing $\delta_{r,u}^{i,t}$ from the state representation is necessary to ensure this user is no more re-selected.

B. Action

The action a_t^i at an step i at TTI t is to select a user $u \in U_r, r \in \mathcal{R}$: $a_t^i = u$ The action space is

$$\mathcal{A} = \Big\{u : \forall r \in \mathcal{R}, u \in \mathcal{U}_r, a_t^j \neq u, \forall j < i, \\ e_t^{r,u} \leq \mu_t^i\Big\}$$

Similar to the above, the action space dimensions decrease after each selection because when a user is selected in the previous action, it should not be allocated twice.

C. Reward

The goal is to optimize fairness. Let $mvalue_t^{r,u}$ be the minimum of the set that consists of the number of transmissions each user of $service^{r,u}$ has made. Then the reward for performing an action $a_t^i = u$ at step i of TTI t when the agent is in state s_t^i is:

$$r_t^i = \begin{cases} tanh(\frac{h_t^{r,u}+1}{T}) & h_t^{r,u} = mvalue_t^{r,u} \\ tanh(\frac{mvalue_t^{r,u}-h_t^{r,u}-1}{T}) & h_t^{r,u} - mvalue_t^{r,u} - 1 \geq 1 \end{cases}$$

The reward is finally multiplied by λ if the user is an eMBB user and by $(1-\lambda)$ if the user is a URLLC user.

D. RL Architecture

We propose to use a neural network that takes as input each $\delta_{r,u}^{i,t}$ and outputs a value for each input. Hence, the same neural network is reused. The outputs of these inputs are fed into a softmax function. The softmax will produce a probability distribution for selecting an action; one user

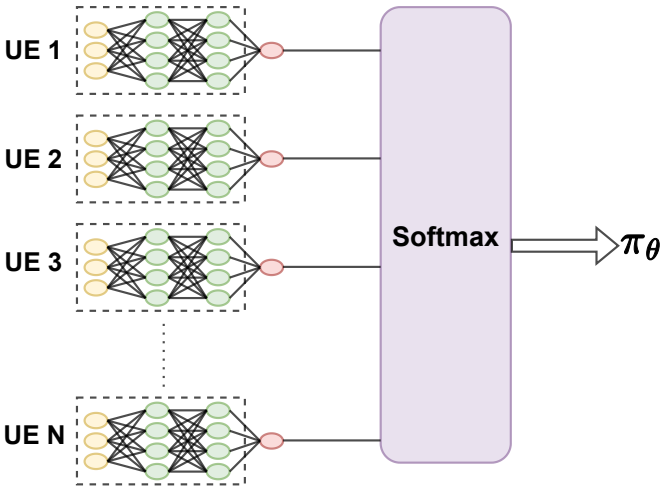


Fig. 2: Neural Network Architecture

will be chosen according to this distribution in each step. In the state representation, only users who can find sufficient computing resources are included. Once the remaining computing resources are insufficient to process any user, the RL agent arrives at the terminal state, and the episode terminates. Fig.2 shows the used architecture. The size of the state is dynamic and depends on the number of users. In case a static architecture is used, a maximum number of users should be defined, and when the number of users is less than the maximum, the remaining neurons will be zero-padded. The latter will require bulking the size of the neural network. The proposed architecture minimizes the neural network's size, reducing the computational requirements. This flexible architecture was also used in [23]. While their RL agent takes just one action per TTI, ours takes one action in a step, and each episode is the collection of decisions made at each step (the selection of a user at each step) in one TTI. The RL model is a policy-gradient model using the algorithm REINFORCE with a Baseline [24]. In Algorithm.1, v_t^i is the normalized discounted reward after subtracting the mean of the rewards of an episode and then dividing by the standard deviation. Using the normalization as a baseline would help stabilize the learning process.

Algorithm 1: RL algorithm

1) initialize users $u \in \mathcal{U}_r$ from all O-RUs $r \in \mathcal{R}$: their MCS, RB, throughput, processing time
2) Initialize weights θ of NN
while *training* **do**
  for $t \in \mathcal{T}$ **do**
    Initialize $i = 1$
    Initialize $\delta_{r,u}^{i,t}$ then s_t^i;
    while s_t^i *is not a terminal state* **do**
      execute a_t^i, get r_t^i, and s_t^{i+1}
      i=i+1
    end
    $\{s_t^i, a_t^i, r_t^i, s_t^{i+1}, a_t^{i+1}, ..., s_t^{end}\} \sim \pi_t$
    calculate v_t^i
    Apply the updates at the end of the episode:
    $\theta \leftarrow \theta + \alpha v_t^i \nabla log_{\pi_t}(s_t^i, a_t^i))$
  end
end

V. Simulation and Results

A. Simulation Environment

To study the performance of the RL Algorithm and compare its performance with respect to the ILP problem, we simulate the environment in MATLAB and use CPLEX for MATLAB to solve the ILP problem. We also use the MATLAB Deep Learning toolbox to model and train the RL agent. We have considered that all O-RUs use 100 RBs, where 90 RBs are used for eMBB users and 10 RBs for URLLC users. The number of RBs per eMBB user follows a uniform random distribution ranging from 20 to 40 RBs, and the range is 1 to 5 for URLLC users. The deadline for eMBB traffic is 2ms as in [13], and the deadline for URLLC is 0.25ms. The number of TTI over which we aim to optimize fairness transmissions among users is 10. So every 10 TTI, users are regenerated in the system, with different MCS and number of RBs. The MCS index is sampled from the real traffic distribution as in [13]. To calculate the throughput, we use the 3GPP specifications to get the Transport Block Size (TBS), and then the value is divided by the TTI duration to determine the throughput [19]. The required processing time is provided in [20] as a function of the MCS, the number of RBs, and the CPU frequency. We set the CPU frequency to 2.6 GHz. Moreover, the prioritization factor λ is set to 0.5. To train the RL, we use a neural network consisting of an input layer with 5 neurons (i.e., the service type is represented using two neurons) and 1 hidden layer consisting of 100 neurons and having the hyperbolic tangent function tanh as an activation function. Using tanh helped our model move faster towards convergence in comparison with sigmoid function. This function is centralized on zero, so it could help speed up the convergence. The learning rate is 0.01, and the discount factor for the rewards is 0.9.

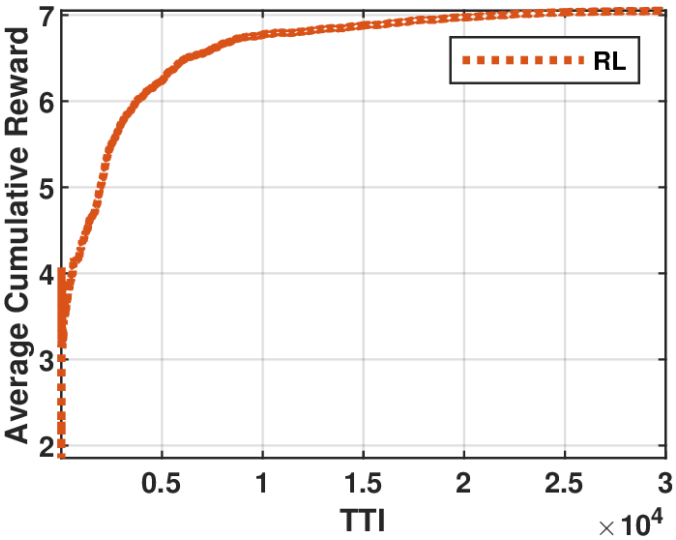


Fig. 3: The average cumulative reward as a function of the number of TTIs

B. RL-Agent Training

To train the RL agent, we suppose that the number of O-RUs that are jointly managed in the O-Cloud follows a random uniform distribution ranging from 5 to 10 O-RUs, where every 10 TTI, this number changes. Increasing the number of O-RUs modifies the total number of users, increasing the computing resources requirements. As Fig.3 shows, the average cumulative reward converges after 25000 TTI.

C. Algorithms Testing

After training the RL-Agent, we test its performance as a function of the number of O-RUs. The performance is compared to the ILP model presentes in Section III-B, Round-Robin (RR), and Proportional Fairness (PF).

The Round-Robin solution tries to allocate users one after the other; if it fails to give resources to a user, it moves to the

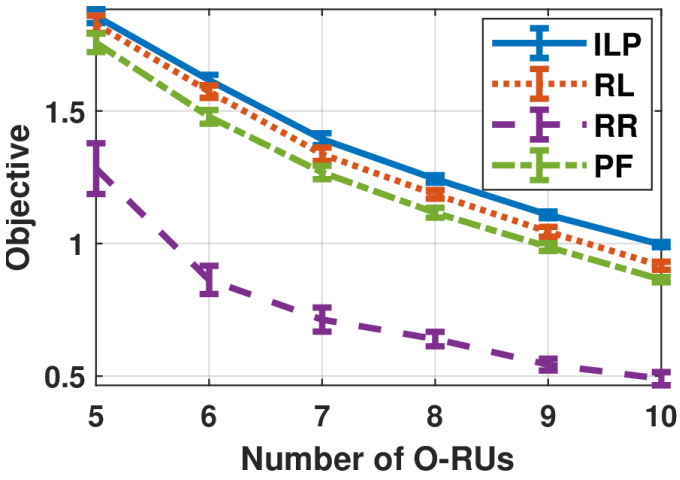


Fig. 4: Objective function value as a function of the number of O-RUs

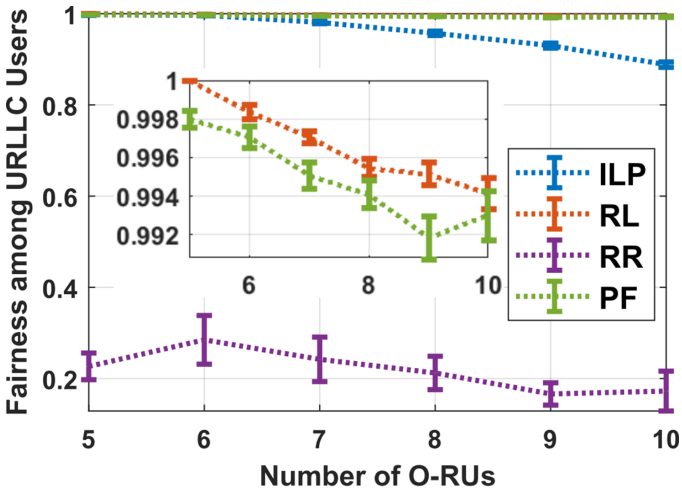


Fig. 6: Fairness among URLLC users as a function of the number of O-RUs

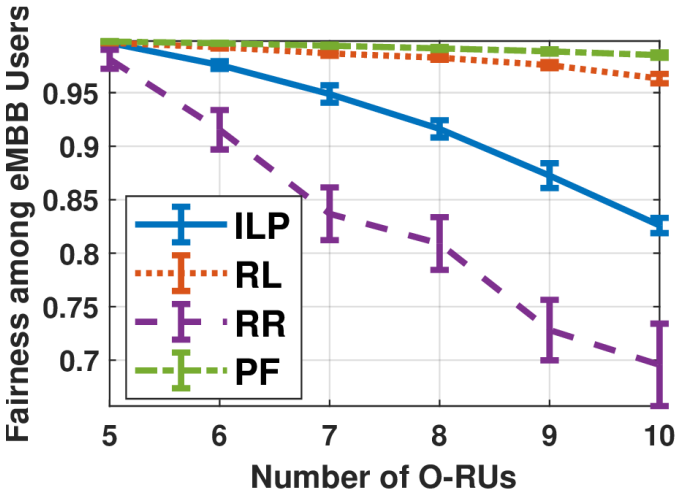


Fig. 5: Fairness among eMBB users as a function of the number of O-RUs

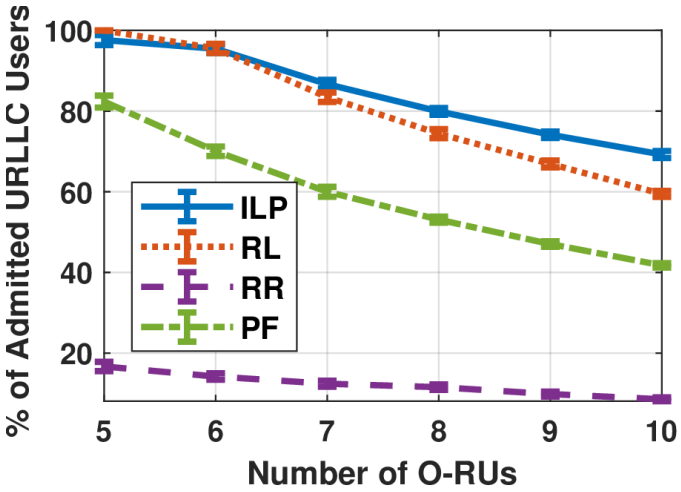


Fig. 7: Percentage of admitted URLLC frames as a function of the number of O-RUs

next one, and it stops if the resources become insufficient to admit any user. At the next TTI, it resumes from where it did the last allocation and repeats.

On the other hand, the Proportional Fairness solution calculates the ratio of achievable throughput if resources are allocated, divided by the historical throughput for each user. Then it picks the user with the highest ratio.

We note that the simulation is repeated for 50 TTIs, and the 95% confidence intervals are computed.

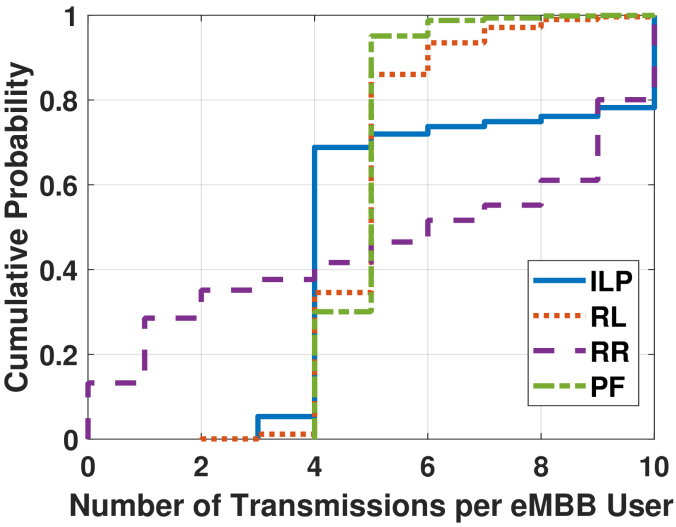
Fig. 4 shows the value of the objective function (i.e., Eq. (1)) for all the four algorithms. Given our goal is to approximate the performance of the ILP solver using RL, the RL agent performs close to the ILP for different numbers of O-RUs, and it is better than PF and much better than RR. In the worst case, the RL will not deviate from the ILP by more than 6%. This shows that the RL agent yields results close to the optimal results of the ILP. On the other hand, we notice that the objective value decreases as the number of O-RUs increases. This can be interpreted by the fact that when the number of O-RUs increase, the number of users increase. This leads to an increase of the demand for computing resources. However, more users in the system means fewer chances of transmission per user. Hence, the objective value decreases as the number of O-RUs increases.

Given that one of our goals is to optimize fairness allocation of resources among eMBB and URLLC users, we plot the curves of Jain's fairness index among eMBB users and among URLLC users in figures 5 and 6 as a function of the number of O-RUs. Fig. 5 shows that the RL algorithm is fairer than the ILP, while RR is the least fair. Recall that the objective is to maximize each user's minimum ratio of transmissions and to try to allocate all the remaining computing resources. Entirely allocating the computing resources will give some users more chances to transmit. However, the fairness metric would decrease as a result. Given that the ILP is the best at

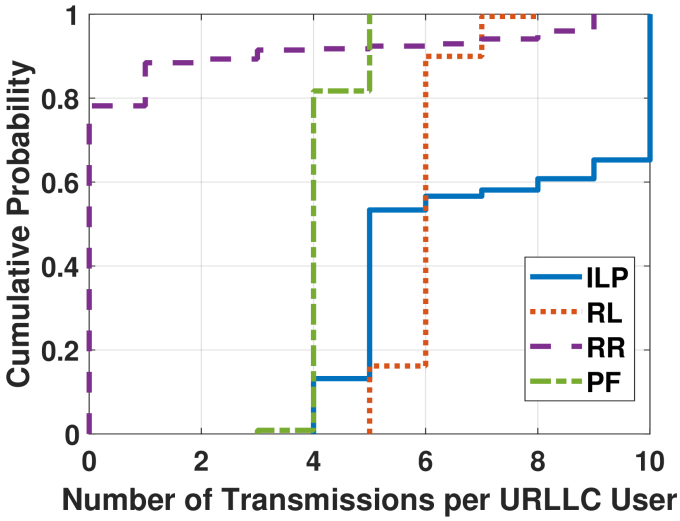
exploiting the computing resources to maximize the objective, it can admit more users, including more eMBB users, which would lower the fairness compared to RL. Additionally, after the ILP maximizes the minimum number of transmissions per user, it aims to maximize the average number of transmissions. This makes the ILP solver free to give some users more chances than others, allowing for more frames to be transmitted. This will likely benefit users with lower processing time requirements, especially URLLC users. Regarding RR, it would give some users more chances than others because of how it allocates users, as explained earlier. Hence the fairness is worsened. Considering the fairness among URLLC users in Fig. 6, the same trend exists as before. However, the RL is almost entirely fair. Due to the existence of eMBB users and the nature of the reward function, the RL agent would prefer to select an eMBB user instead of URLLC when there are little remaining resources. This helps keep the fairness level close to 1. For the same reason explained earlier, the RL outperforms the ILP regarding fairness among URLLC users. On the other hand, the RR does not detect the difference in deadline requirements for URLLC, so it fails to allocate them fairly. As mentioned earlier, the decrease in fairness when the number of O-RUs increases is due to having more users in the system, and this will make some users have more transmissions and degrade fairness. As figures 5 and 6 show, PF is very close to RL; it is better for eMBB but slightly lower for URLLC. The reason is that the PF algorithm is oblivious to the stricter deadline requirements of URLLC users. This would lead to having more resources given to eMBB users.

While the fairness among URLLC users is very close for the RL and the PF algorithms, the number of URLLC frames that the RL algorithm admits is higher than the PF does by about 20%, as Fig. 7 shows. This means that while each algorithm has similar transmission opportunities for URLLC users leading to similar fairness scores, the RL algorithm can elevate the number of transmissions for URLLC users. This makes the RL suitable when the goal is to allow more URLLC frames to be transmitted without entirely sacrificing eMBB users.

Fig. 8 shows the cumulative distribution of the number of transmissions that eMBB and URLLC users get when the number of O-RUs is 10. We recall that we suppose that each user persists in the system for 10 TTIs so that a user may transmit at a maximum of 10 frames. As Figures 8a and 8b show, RL and PF curves have close trends. For instance, for eMBB users, the 80^{th} percentile is equal to 5 transmissions in both RL and PF algorithms while for URLLC it is 4 transmissions for PF and 6 transmissions for RL; this justifies why they have similar fairness scores, given that the majority of users will get the same transmissions, which would elevate fairness, as explained in previous sections. On the other hand, the ILP is more flexible; once it allocates a minimum for all users, it gives users varied chances. As described earlier, RR fails to admit a lot of URLLC users, and eMBB users have varying transmission chances. This justifies why it is the worst concerning the fairness metric. We recall that fairness is maximized when all users get an equal number of transmissions.



(a) Cumulative distribution function of the number of transmissions for eMBB users



(b) Cumulative distribution function of the number of transmissions for URLLC users

Fig. 8: Cumulative Probability Distribution for the number of transmissions that users get when the number of O-RUs is 10

To summarize, the RL and PF algorithms are the fairest for both eMBB and URLLC users. However, RL is more suitable when the operator prefers to give more chances to URLLC users. Again, we recall that URLLC users should be able to achieve very low latency and have high reliability; however, we are considering the case where the available resources suddenly become insufficient, and at the same time, the loss of some URLLC frames is non-fatal. Hence the operator can try to optimize the allocation of computing resources to URLLC users and, at the same time, not neglect eMBB users.

VI. Conclusion

O-RAN opens the path for deploying Reinforcement Learning based algorithms to execute different tasks. In this paper, we have considered the problem of allocating computing resources to eMBB and URLLC users from multiple O-RUs. We have formulated an ILP-based optimization problem that aims to optimize fairness allocation among eMBB and URLLC users and improve the QoS guarantees for URLLC users without neglecting eMBB users. Due to the NP-hardness of solving the ILP problem, we propose an RL-based algorithm. Simulation results show the ability of an RL agent to perform close to the optimal ILP solver, optimizing the performance of URLLC users without neglecting eMBB users. Our results demonstrate that the objective value does not deviate from the ILP objective by more than 6%. Additionally, it provides high fairness for eMBB and URLLC users while admitting more URLLC users in comparison to the known Proportional Fairness algorithm. For future work, we aim to study the benefits of using Federated Learning for our RL problem.

Acknowledgment

This work was funded by the ANR HEIDIS (https://heidis.roc.cnam.fr/; ANR-21-CE25-0019) project.

References

[1] M. Z. Chowdhury, M. Shahjalal, S. Ahmed, and Y. M. Jang, "6g wireless communication systems: Applications, requirements, technologies, challenges, and research directions," *IEEE Open Journal of the Communications Society*, vol. 1, pp. 957–975, 2020.

[2] L. Bonati, S. D'Oro, M. Polese, S. Basagni, and T. Melodia, "Intelligence and learning in o-ran for data-driven nextg cellular networks," *IEEE Communications Magazine*, vol. 59, no. 10, pp. 21–27, 2021.

[3] A. K. U and G. Gundu Hallur, "Economic and technical implications of implementation of openran by "rakuten mobile"," in *2022 International Conference on Decision Aid Sciences and Applications (DASA)*, 2022, pp. 959–964.

[4] S. Husain, A. Kunz, A. Prasad, E. Pateromichelakis, K. Samdanis, and J. Song, "The road to 5g v2x: Ultra-high reliable communications," in *2018 IEEE Conference on Standards for Communications and Networking (CSCN)*, 2018, pp. 1–6.

[5] F. Salah, L. Kuru, and R. Jäntti, "Reliability and availability enhancements of the 5g connectivity for factory automation," in *2019 IEEE 17th International Conference on Industrial Informatics (INDIN)*, vol. 1, 2019, pp. 1027–1032.

[6] D. Avidor, S. Mukherjee, J. Ling, and C. Papadias, "On some properties of the proportional fair scheduling policy," in *2004 IEEE 15th International Symposium on Personal, Indoor and Mobile Radio Communications (IEEE Cat. No.04TH8754)*, vol. 2, 2004, pp. 853–858 Vol.2.

[7] M. Polese, L. Bonati, S. D'Oro, S. Basagni, and T. Melodia, "Understanding O-RAN: architecture, interfaces, algorithms, security, and research challenges," *CoRR*, vol. abs/2202.01032, 2022. [Online]. Available: https://arxiv.org/abs/2202.01032

[8] S. D'Oro, L. Bonati, M. Polese, and T. Melodia, "Orchestran: Network automation through orchestrated intelligence in the open RAN," *CoRR*, vol. abs/2201.05632, 2022. [Online]. Available: https://arxiv.org/abs/2201.05632

[9] Y. Shi, Y. E. Sagduyu, and T. Erpek, "Reinforcement learning for dynamic resource optimization in 5g radio access network slicing," in *2020 IEEE 25th International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*, 2020, pp. 1–6.

[10] J. Mei, X. Wang, K. Zheng, G. Boudreau, A. B. Sediq, and H. Abou-Zeid, "Intelligent radio access network slicing for service provisioning in 6g: A hierarchical deep reinforcement learning approach," *IEEE Transactions on Communications*, vol. 69, no. 9, pp. 6063–6078, 2021.

[11] F. W. Murti, S. Ali, and M. Latva-aho, "Deep reinforcement based optimization of function splitting in virtualized radio access networks," in *2021 IEEE International Conference on Communications Workshops (ICC Workshops)*, 2021, pp. 1–6.

[12] M. Elsayed and M. Erol-Kantarci, "Ai-enabled radio resource allocation in 5g for urllc and embb users," in *2019 IEEE 2nd 5G World Forum (5GWF)*, 2019, pp. 590–595.

[13] M. Sharara, S. Hoteit, P. Brown, and V. Vèque, "Coordination between Radio and Computing Schedulers in Cloud-RAN," in *2021 IFIP/IEEE International Symposium on Integrated Network Management (IM)*.

[14] M. Sharara, S. Hoteit, and V. Vèque, "A Recurrent Neural Network Based Approach for Coordinating Radio and Computing Resources Allocation in Cloud-RAN," in *2021 IEEE 22nd International Conference on High Performance Switching and Routing (HPSR)*.

[15] O-RAN-WG1, "Slicing Architecture," pp. 1–53, 2020.

[16] O-RAN-WG6, "Cloud Architecture and Deployment Scenarios for O-RAN Virtualized RAN," pp. 1–46, 2019.

[17] O-RAN-WG1, "Architecture Description," pp. 1–36, 2021.

[18] S. D'Oro, M. Polese, L. Bonati, H. Cheng, and T. Melodia, "dapps: Distributed applications for real-time inference and control in o-ran," 2022. [Online]. Available: https://arxiv.org/abs/2203.02370

[19] (2018, October) 5G; NR; Physical layer procedures for data, ETSI TS 138 214 V15.3.0.

[20] S. Khatibi, K. Shah, and M. Roshdi, "Modelling of computational resources for 5g ran," in *2018 European Conference on Networks and Communications (EuCNC)*, 2018, pp. 1–5.

[21] R. Jain, D. Chiu, and W. Hawe, *A Quantitative Measure of Fairness and Discrimination for Resource Allocation in Shared Computer Systems*. DEC Research Report TR-301, Sep 1984.

[22] B. Korte and J. Vygen, *Combinatorial Optimization: Theory and Algorithms*, 5th ed. Springer Publishing Company, Incorporated, 2012.

[23] J. S. Shekhawat, R. Agrawal, K. G. Shenoy, and R. Shashidhara, "A reinforcement learning framework for qos-driven radio resource scheduler," in *GLOBECOM 2020 - 2020 IEEE Global Communications Conference*, 2020, pp. 1–7.

[24] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: A Bradford Book, 2018.