Percentile-Based Deep Reinforcement Learning and Reward Based Personalization For Delay Aware RAN Slicing in O-RAN

Peyman Tehrani [1] **Anas Alsoliman** [1]

Abstract

In this paper, we tackle the challenge of radio access network (RAN) slicing within an open RAN (O-RAN) architecture. Our focus centers on a network that includes multiple mobile virtual network operators (MVNOs) competing for physical resource blocks (PRBs) with the goal of meeting probabilistic delay upper bound constraints for their clients while minimizing PRB utilization. Initially, we derive a reward function based on the law of large numbers (LLN), then implement practical modifications to adapt it for real-world experimental scenarios. We then propose our solution, the Percentile-based Delay-Aware Deep Reinforcement Learning (PDA-DRL), which demonstrates its superiority over several baselines, including DRL models optimized for average delay constraints, by achieving a 38% reduction in resultant average delay. Furthermore, we delve into the issue of model weight sharing among multiple MVNOs to develop a robust personalized model. We introduce a reward-based personalization method where each agent prioritizes other agents' model weights based on their performance. This technique surpasses traditional aggregation methods, such as federated averaging, and strategies reliant on traffic patterns and model weight distance similarities.

1. Introduction

1.1. Motivation

The field of wireless communication has seen tremendous advancements in recent years, leading to the emergence of 5G networks that promise to offer unprecedented speeds, reliability, and capacity. However, with the increasing number of devices and applications that require connectivity, the demand for more efficient and flexible radio access network (RAN) management has become a pressing issue. Thus, mobile network operators are expected to support diverse range of applications where each of them has its own specifc requirements and constraints. These applications range from mission-critical applications like autonomous driving and remote surgery, necessitating high reliability and low latency communication links, to high-bandwidth demanding applications such as virtual reality and 4K video streaming. Furthermore, the rise of internet of things (IoT) and smart city initiatives demands networks that can handle long-range communications with numerous active devices. In response to the escalating demands of evolving technologies, mobile network operators (MNO) are veering toward solutions like network and RAN slicing (Wijethilaka & Liyanage, 2021; Elayoubi et al., 2019). These methods enable the partitioning of a single network into multiple tailored network slices, ensuring each segment meets the specific requirements of its use-case. This tailored approach not only reduces operational costs but significantly improves the overall quality of service (QoS).

Network and RAN slicing is a well studied problem in recent years, as many work considered optimization and model based approachs (Motalleb et al., 2022; Popovski et al., 2018). However, the industry's shift towards open radio access networks (O-RAN) (Polese et al., 2023; Bonati et al., 2021b) can provide new possibilities to the field. Integrating artificial intelligence (AI) and machine learning (ML), particularly in the RAN intelligent controller (RIC) component, paves the way for a data-driven, closed-loop control system. Utilizing the vast amounts of data generated in the network every second, these data-driven approaches can significantly boost the performance and energy efficiency of the future generation of network infrastructure. Unlike the static model based methods, the learning-based approaches offer dynamic adaptability to network environments, enabling MNOs to develop specialized ML models. These models can be tailored to specific regions and use cases, taking into account unique factors such as user traffic patterns, level of QoS, wireless propagation of geographical areas, and other relevant environmental variables.

[1] Donald Bren School of Information and Computer Sciences, University of California at Irvine, United States. Correspondence to: Peyman Tehrani <peymant@uci.edu>.

Proceedings of the 42^{nd} International Conference on Machine Learning, Vancouver, Canada. PMLR 267, 2025. Copyright 2025 by the author(s).

1.2. Main Contributions

In this paper we study the RAN slicing problem in an O-RAN environment. Specifically, we consider a physical resource block (PRB) allocation problem for multiple mobile virtual network operators (MVNOs) in a heterogeneous setting, where the client traffic demands, number of users, required QoS, and wireless propagation environment for each MVNO could be completely different among the other MVNOs. The complexity and diversity of these environmental variables necessitate a specialized model capable of dynamically adapting to each MVNO's setting. To address this, we utilize a data-driven approach, specifically employing deep reinforcement learning (DRL) algorithms. These algorithms have shown notable efficacy in managing complex control tasks in challenging settings (Mnih et al., 2015). Additionally, we propose a reward function based on the law of large numbers (LLN) to satisfy the probabilistic upper bound delay constraint of the MVNOs's clients data request.

We propose a Percentile-based Delay Aware DRL (PDA-DRL) solution and demonstrate its superiority over a diverse set of baselines. We further explore how multiple MVNOs can collaboratively share their model weights without disclosing any user data to improve their policy performance. We demonstrate that federated learning baselines such as federated averaging (FedAV) (McMahan et al., 2017) will not work in this setting as there is a huge difference between the environments that each agent interacts with. Instead, we propose a reward-based personalization method in which each agent up-weights the other agents' models based on their estimated performance on their own environment. We also validate our approach through online DRL experiments using our experimental framework, which incorporates the Colosseum, the world's largest RF emulator (Bonati et al., 2021c), and Scope, a next-generation wireless network prototyping platform (Bonati et al., 2021a).

In summary, our contributions are multifaceted and distinct from other works reviewed in Section 2, and can be summarized as follows:

- **Percentile-Based Delay Guarantees**: Our work explores percentile-based delay guarantees as a more robust metric than the average delay or throughput, which are commonly discussed in the literature.
- **Reward Shaping Function**: We introduce a distinctive reward shaping function, built on top of the RAN slicing constrained optimization solution.
- **Performance Based Personalization Method**: Our personalization approach is novel, focusing on the performance of agents rather than similarities like traffic patterns or model weights. With the increasing accessibility of digital twins (Ericsson, 2023; Sun et al., 2024b) and generative AI (Karapantelakis et al., 2023), our proposed method would be practical and implementable by network operators.
- **Validation on Experimental Testbed**: Unlike most works in Section 2 that rely solely on simulation, we validate our DRL solution using the Colosseum experimental testbed.

2. Literature Review

Recently, the deployment of ML in O-RAN has garnered significant attention for a variety of use cases, including energy saving (Akman et al., 2024), traffic steering (Lacava et al., 2022), resource allocation (Joda et al., 2022), automation (D'Oro et al., 2022), traffic prediction (Niknam et al., 2022), anomaly detection (Sun et al., 2024a), access control (Cao et al., 2021), load balancing (Orhan et al., 2021), remote electrical tilt optimization (Vannella et al., 2022), (Orhan et al., 2021), admission control (Lien et al., 2021), energy efficiency (Kalntis & Iosifidis, 2022), security (Xavier et al., 2023), spectrum sensing (Reus-Muns et al., 2023), virtual reality (Kougioumtzidis et al., 2023), MU-MIMO interference coordination (Ge et al., 2023) and many others. In the context of RAN slicing, several notable works have been conducted and here we will provide a review of the most related studies.

Bakri et al. (Bakri et al., 2021) utilize traditional ML algorithms like support vector machines for predicting radio resource requirements in network slices, based on user channel quality feedback. Filali et al. (Filali et al., 2022) propose a two-level RAN slicing method within the O-RAN framework, focusing on the allocation of communication and computation resources using a double deep Q-network (DDQN) algorithm. Yang et al. (Yang et al., 2021) address RAN slicing for IoT and URLLC services, formulating it as an optimization problem and proposing a solution via the alternating direction method of multipliers (ADMM). Hua et al. (Hua et al., 2019) introduce a generative adversarial network-powered deep distributional Q Network (GAN-DDQN) for resource management in network slicing. Setayesh et al. (Setayesh et al., 2022) and Wu et al. (Wu et al., 2020) explore multi-timescale RAN slicing problems, proposing hierarchical deep learning frameworks and studying RAN slicing for Internet of Vehicles (IoV) services, respectively. Mei et al. (Mei et al., 2021) focus on a two-layered RAN slicing strategy aimed at maximizing QoS and spectrum efficiency.

However, none of the previous works consider a percentile-based delay guarantee or probabilistic QoS constraint. Moreover, the aforementioned studies primarily focus on single wireless environments and do not address collaborative efforts among different agents in varied and diverse environ-

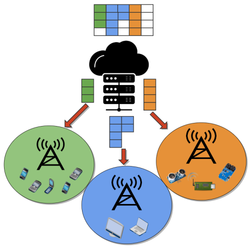


Figure 1: A network containing a central controller (telco operator) and multiple RANs with their associated users.

ments and tasks. Few works, such as (Messaoud et al., 2020; Tehrani et al., 2021; Abouaomar et al., 2022) has studied aggregating multiple DRL agents. Nonetheless, these approaches mainly employ federated averaging (McMahan et al., 2017), which, as our study demonstrates, is less effective in environments with significant variations in traffic patterns, wireless propagation, and task-specific QoS levels.

Among the limited studies that focus on personalization for each DRL agent's model, Rezazadeh et al. (Rezazadeh et al., 2022) propose a dynamic user clustering method based on similarities in local traffic conditions. However, their approach does not account for service type and QoS level, reducing its applicability in scenarios with diverse QoS constraints. Nagib et al. (Nagib et al., 2023) explore the use of transfer learning to accelerate RL-based RAN slicing convergence, suggesting a predictive method to choose the most appropriate pre-trained policy by focusing on convergence error and weight vector distances. In contrast, our research points out the inadequacies of relying solely on model weight similarities, demonstrating that personalization based on model performance is more effective.

3. System Model

We consider a network composed of a core network (CN) and multiple RANs, each RAN is represented as a cell tower providing radio access to CN. We assume that both the CN and the RANs are owned and managed by a single telco operator (TO) which provides radio network resources, represented as PRBs, to MVNOs, who periodically rent a slice of RAN resources to provide cellular network access to their own mobile users as illustrated in Fig. 1. Therefore in each cell, multiple MVNOs would compete for physical RAN resources based on their clients demand and the QoS that they have guaranteed to their users.

To be more precise, assume that there exists K different RAN clusters (or we can call it cell) which we index it with $k \in \mathcal{K} = \{1, 2, ..., K\}$ and there are total of V MVNOs as we index it by $v \in \mathcal{V} = \{1, 2, ..., V\}$. We show the user j in cell k which is client of MVNO v as j_k^v. We also assume each MVNO has its own set of strict QoS requirements. Here, we focus on maximum transmission latency as a QoS metric perceived by the operator's clients. We denote the maximum transmission latency guaranteed by the vth MVNO as D_{max}^v.

We also assume each user has a specific data request distribution, where the number of requests for user j in cell k is drawn from $Q_{k,j}$ and the data size from $F_{j,k}$. For example, sensory clients generate many small packets, while large file downloads result in fewer, larger packets. These distributions vary based on protocols, user types, and timing, and the request arrivals may not follow predefined models known to the MVNOs.

At each slicing time slot T_S, each MVNO determines the number of PRBs required to meet its users' demands, in line with anticipated QoS. The quantity of requested PRBs depends on factors like latency requirements, wireless channel quality, data request queue length, the monetary value of resource blocks, and other environmental variables. Here, PRB refers to a time-frequency network resource block, with a time duration of one PRB being $T_B[s]$ time-width and $W[Hz]$ frequency-width. We also define two distinct timing scales: the PRB time slot T_B and the slicing time slot T_S. The T_S is greater than T_B and we assume it is an integer multiplicative of T_B as $T_S = HT_B$. In our slicing framework, at the start of each T_S, each MVNO requests a specific amount of PRBs from TO. Upon allocation, a scheduling algorithm distributes the assigned PRBs to mobile users at the T_B time scale in order to meet the required QoS.

Due to the nature of the queuing and existence of unknown traffic patterns, and the fact that current actions directly affect future states and actions, this problem is a sequential decision-making problem. One appropriate tool for solving this type of problem is DRL. Essentially, we can model the problem as a two-layer optimization problem. At the higher level, the MVNO acts as an DRL agent that needs to decide at every T_S seconds, how many PRBs to request from the TO in order to satisfy both its near-future traffic demands and its monetary budget. At the lower level, the MVNO must satisfy its users' latency requirements over the next HT_B time slots by solving a scheduling problem using the assigned PRBs. This procedure is summarized in Fig. 2.

4. Problem Formulation

The downlink SNR of user j in cell k at time slot t can be defined as:

$$\gamma_{j,k}^t = \frac{P_k^t g_{j,k}^t}{N_j} \tag{1}$$

where $g_{j,k}^t$ is the channel gain between the kth RAN and jth user at time slot t, P_t is the kth RAN's RF transmit

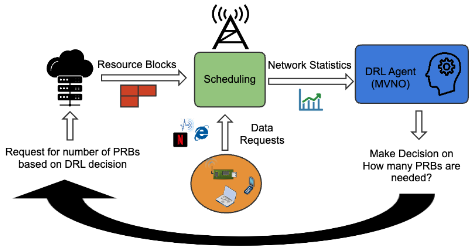

Resource Blocks
Scheduling
Network Statistics
DRL Agent (MVNO)
Data Requests
Request for number of PRBs based on DRL decision
Make Decision on How many PRBs are needed?

Figure 2: RL scheme for the RAN slicing problem.

power and N_j is the noise power at the user j terminal. It is assumed that the each RAN is using uniform power allocation for all PRBs. We also assume that the channel gain $g^t_{j,k}$ includes both small-scale and large-scale fading factors, and it remains constant for a period of length T_b.

Each RAN use a predefined mapping lookup table based on the channel quality indicator (CQI) reported by the mobile users, which maps the measured SNR at user j to an integer bit rate capacity which is related to a modulation and coding scheme (MCS) as follows $\Psi : \mathcal{R} \rightarrow \mathcal{N}$. The number of deliverable bits for the jth user request in the requests' buffer for kth PRB at time t would be:

$$C^t_{j,k} = WT_b\Psi(\gamma^t_{j,k}) \tag{2}$$

We define $f^t_{j^v_k,q}$, as the qth packet size requested in cell k by user j for operator v at time t. At each PRB time slot, $C^t_{j,k} l^t_{j^v_k,q}$ bits will be delivered to user j, where $l^t_{j^v_k,q}$ is the number of PRBs allocated to the qth request of user j by operator v. Therefore given that the guaranteed transmission latency for vth operator is D^v_{max}, then the following constraint should be satisfied:

$$\sum_{t=t_0}^{t_0+d_{j^v_k,q}} C^t_{j,k} l^t_{j^v_k,q} \geq f^{t_0}_{j^v_k,q} \tag{3}$$

$$d_{j^v_k,q} \leq D^v_{max} \tag{4}$$

where $d_{j^v_k,q}$ is the transmission delay for qth packet requested in cell k by user j for operator v. Basically, constraint (3) ensures that entire data, in terms of number of bits, is delivered and constraint (4) implies that that transmission latency should not violate the D^v_{max} which was guaranteed by the MVNO.

Given the fixed number of PRBs at each T_s, and the stochastic and non-stationary nature of users' requests in each cell, it would be challenging to satisfy all users' demands with a %100 guarantee unless each MVNO requests extra PRBs at each network slicing time slot, which is not cost-efficient. Therefore, our goal here is to design a controller that guarantees a probabilistic upper-bound on the user delay requirement, i.e., $Pr(d_{j^v_k,q} > D^v_{max}) < \epsilon_v$, with the minimum possible PRBs to satisfy such requirement.

Mathematically speaking, the sequential decision making problem that each MVNO has to solve can be formulated as follows:

$$\min \lim_{T\to\infty} \frac{1}{T} \sum_{T_s=1}^{T} N^v_{T_s} = \mathbb{E}\left\{N^v_{T_s}\right\} \tag{5}$$

$$\text{s.t.} \quad Pr(d_{j^v_k,q} > D^v_{max}) < \epsilon_v \quad \forall k,j,q,v \tag{6}$$

where $N^v_{T_s}$ is the number of PRBs used in time slot T_s for vth MVNO's clients which is equal to:

$$N^v_{T_s} = \sum_{t=t_0}^{t_0+T_s} \sum_j \sum_q l^t_{j^v_k,q} \tag{7}$$

So the goal is to minimize the expected number of PRBs used over an infinite horizons of network slicing time slots, while guaranteeing that dissatisfactions of users occurs with less than ϵ_v probability as reflected in (6).

5. Deep Reinforcement Learning

In this section, we will discuss the design of the MVNO's slice controller and how to represent the controller's logic as a reinforcement learning agent. In a reinforcement learning problem, the agent iteratively interacts with the environment. This environment is usually described by a Markov decision process (MDP) which can be defined with 5-tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$, which is state space $\mathcal{S}$, action space $\mathcal{A}$, state transition probability $Pr(s_{t+1}|s_t, a_t) \in \mathcal{P}$, reward function R, and discount factor $\gamma \in \big[0, 1\big)$. According to this notation, at each time t, the agent based on its current state s_t takes an action $a_t \in \mathcal{A}$, and goes from state s_t to a new state s_{t+1} with probability $Pr\big(s_{t+1}|s_t, a_t\big)$ and receives reward of r_{t+1} from the environment. If we define policy $\pi\big(s, a\big)$ as the probability of taking action $a_t = a$ in state $s_t = s$, i.e, $\pi\big(s, a\big) = Pr\big(a_t = a|s_t = s\big)$, the goal of the agent is to learn a policy that maximizes the expected sum of discounted rewards it receives over the long run. This sum of discounted rewards is called 'return' and is defined as $R_t = \sum_{i=0}^{T} \gamma^{i+t} r_{t+i}$ where T is the length of the time horizon. We define the optimal policy π^*, that chooses best actions a^* in state s, which maximizes the expected return without knowledge of the function form of the reward and the state transitions.

$$\pi^* = \underset{\pi}{\operatorname{argmax}}\, E_\pi\{R|s_0 = s\} \tag{8}$$

In order to solve the optimization problem (5) using DRL, first we need to define the problem in a RL setting. To this end, we need to define the states, actions and rewards for the MVNOs which are the RL agents in our context.

5.1. States

Once the number of PRBs has been determined, the RAN's scheduling algorithm allocates the available PRBs to incoming requests on the T_b time scale. We consider various network statistics as input state to our DRL model. These features include the percentage of requests that meet the delay deadline, average delay, standard deviation of all requests in the current network slicing time slot, average and standard deviation of SNR at the client side (average PRB capacity) which takes into account the wireless propagation environment and the mobility of users, average demand-to-capacity ratio (i.e., average packet size request divided by capacity), and the number of allocated PRBs in the previous time slot. To track network behavior and user traffic patterns, these features are considered over a history window of length h, rather than focusing solely on current time slot values. This approach builds a stronger model capable of capturing the temporal correlation of these features.

5.2. Actions

In our setting, actions determine the proper number of PRBs given the current state. This could be represented by a finite integer set, such as $\mathcal{A} = \big\{10, 20, 30, ..., 120\big\}$, assuming the maximum available PRBs for the entire RAN are 120. However, a more reasonable approach, which we have chosen for our setting, involves using differential actions due to the strong temporal correlation observed in PRB utilization in real-world environments. This approach prompts the DRL agent to decide on the increase or decrease in the number of PRBs for the next network slicing time slot. Thus, a possible action space in this scenario could be:

$$\mathcal{A} = \big\{-2^J, -2^{J-1}, ..., -1, 0, 1, ..., 2^{J-1}, 2^J\big\} \quad (9)$$

where the $|\mathcal{A}| = 2J + 1$

5.3. Reward

Optimal reward function could based be derived based on the optimization problem (5) and constraint (6) as below:

$$J_{\pi_\theta} = L_{\theta,\lambda} = -\mathbb{E}\big\{N_{T_s}\big\} + \lambda(Pr(d_q < D^{max}) - (1-\epsilon)) \quad (10)$$

As shown Appendix A, maximizing the average reward J_{π_θ} with respect to θ is equivalent to computing the Lagrangian dual function associated with problem (5).

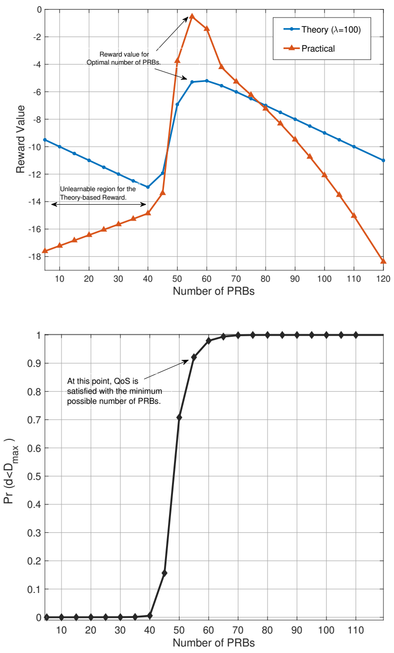


Figure 3: (a) Theory vs practical reward functions. (b) Probability of satisfying of the maximum delay constraint.

6. Practical Modification of Reward Function

We can derive an optimization-based reward function as shown in previous section and Appendix A. However, one problem that exists is that, in theory, it is assumed that the second term in the reward equation (10) always increases with respect to N_{Ts}, as we expect that by increasing the number of PRBs, the probability of violating the delay constraint decreases.

However, in reality as we see in Fig. 3a, there is a threshold below which none of the packets satisfy the delay constraint. In practice, what occurs is that increasing the PRBs decreases the average delay; however, there might still be instances where none of the packet transmission delays fall below D^{max}, leading to a zero probability of satisfaction. Consequently, the only dominant part of the reward function, $-N_{Ts}$, would motivate the agent to decrease the requested PRBs and this would make the agent training very challenging and almost impossible. We have marked this region in Fig. 3a as an "unlearnable region" because training an agent using gradient descent-based algorithms with initial conditions in this region would fail, as the agent would eventually request zero PRBs.

Therefore, in order to have a practical reward function which could be used in different environment conditions and also provide a more robust and faster learning convergence, we propose the below modified reward function:

$$r = \begin{cases} -\Delta\gamma_p + e^{\zeta_p \Delta + \nu_p} N_{T_s}^2 & \text{if } \Delta \geq 0 \\ \Delta\gamma_n + e^{\zeta_n \Delta + \nu_n} N_{T_s} & \text{if } \Delta < 0 \end{cases}$$

where

$$\Delta = Pr(d_q < D^{max}) - (1-\epsilon) \tag{11}$$

This new reward function is composed of two sections. When the constraint is not satisfied ($\Delta < 0$), the first term of the reward function is similar to the second term of equation (22) as an increase in the probability of satisfaction contributes linearly to the increase in the reward value, while the second term is linear in N_{T_s} with an exponential coefficient that depends on Δ itself and the coefficients ζ_n and ν_n. The ζ_n and ν_n can be set in a way that, when Δ is negative, the exponential term is large and motivates the agent to increase N_{T_s} faster. As Δ approaches zero, the exponential term fades, and the linear term becomes dominant. Similar analysis holds for the case when $\Delta > 0$.

As observed in Fig. 3a, both the original and modified reward functions reach their maximum at the same point (PRB=55), where the probabilistic constraint is satisfied with the minimum number of PRBs (as shown in Figure 3b for $\epsilon = 0.1$). Therefore, the modified reward function achieves the same optimal point while being shaped in a way that promotes more robust and faster agent training. It is important to note that, for convergence guarantee, we clip the output of the function to be bounded between $(-R_{max}, 0)$.

7. DRL Personalization

After the training phase, each agent associated with a particular MVNO in a cell would develop its own DRL model. However, we want to further enhance the performance of these models by leveraging the knowledge and expertise of other agents. This is where personalized federated learning comes into play. Personalized federated learning (Huang et al., 2021) offers a key benefit of creating highly personalized models for each agent without compromising data privacy. By combining the knowledge and models from multiple agents, we can generate a more robust and accurate model tailored for a specific MVNO and RAN.

Mathematically speaking, considering N DRL agents with a model weights vector $\mathbf{W} = [W_0, W_1, ..., W_{N-1}]$, each agent i aims to find the optimal aggregation coefficient vector $\boldsymbol{\alpha_i}$. This vector is designed to maximize the performance of the resulting personalized model W_i^p, as described in (12), when executed in its own environment i.

$$W_i^p = \sum_{j=1}^{N} \alpha_{i,j} W_j \tag{12}$$

Here, we propose a reward-based personalization for DRL models. The main idea behind this method is that each agent utilizes the model weights of other agents based on their respective performance. Additionally, we will compare this approach with other model aggregation methods such as those considering similarities in model weight (Nagib et al., 2023) and traffic patterns and wireless environment features (Rezazadeh et al., 2022) and also model averaging (Tehrani et al., 2021).

7.1. Personalizing Based on Feature Weights

In this method, each agent would aggregate other agents' models based on a feature vector that represent the wireless environment and QoS of the associated RAN and MVNO respectively. We define the feature vector of the ith agent model as $\mathbf{f_i} = \left[D_{max_i}, \epsilon_i, \tilde{K}_i, \tilde{C}_i, \tilde{T}_i\right]$, where D_{max_i} and ϵ_i are related to QoS of ith agent, $\tilde{K}_i$, $\tilde{C}_i$ and $\tilde{T}_i$ represent the average number of users, average channel capacity and average number of requested data packets for the ith environment respectively. In order to compute distances of two feature vectors, we use the following distance function:

$$dist(\mathbf{f_i}, \mathbf{f_j}) = \sum_m \exp \frac{\|f_i^m - f_j^m\|^2}{\sigma_m} \tag{13}$$

where σ_m is a temperature parameter and could be proportional to the standard deviation of mth feature. The aggregation coefficient $\alpha_{i,j}$ can be obtained as follows:

$$\alpha_{i,j} = \frac{dist(\mathbf{f_i}, \mathbf{f_j})}{\sum_{j=1}^{N} dist(\mathbf{f_i}, \mathbf{f_j})} \tag{14}$$

7.2. Personalizing Based on Model Weights

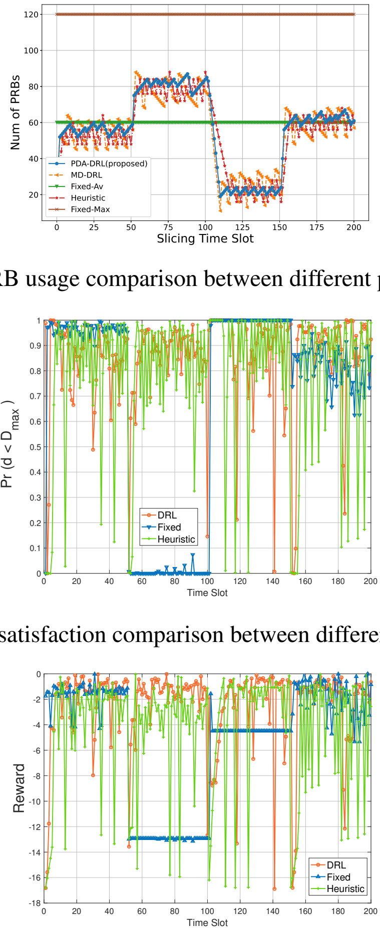
In this method, each agent would obtain the aggregation coefficient based on the similarity of the DRL model weights. Here we use L_2 norm as distance function in this approach:

$$\alpha_{i,j} = \frac{\|W_i - W_j\|^2}{\sum_{j=1}^{N} \|W_i - W_j\|^2} \tag{15}$$

7.3. Reward Based Personalization

In this method, the ith agent computes the aggregation coefficients for other agents' models based on their performance in its own environment. Agent i tests model W_j on its own environment i for T episodes and obtains the average reward $\hat{R}_{i,j}^T$. The aggregation coefficient is then obtained as follows:

$$\alpha_{i,j} = \frac{e^{\beta \hat{R}_{i,j}^T}}{\sum_j e^{\beta \hat{R}_{i,j}^T}} \tag{16}$$



(a) PRB usage comparison between different policies.

(b) QoS satisfaction comparison between different policies.

(c) Reward comparison between different policies.

Figure 4: Performance comparison of DRL-based methods, heuristic and fixed polices when there is a dynamic user traffic with increasing and decreasing patterns.

where β is a temperature parameter, the algorithm behaves similarly to federated averaging when β is close to zero. As β increases, the algorithm prioritizes selecting the best model.

In Appendix B we discussed how the training and personalization of different models for different RANs would be aligned in O-RAN compliance architecture.

8. Simulation Results

In this section, we define the simulation scenarios, specify the parameters, and provide a comprehensive performance evaluation of our proposed algorithms. The DRL models have been implemented in PyTorch, and we used the Adam optimizer with a learning rate of $lr = 0.0003$ for training. Our chosen DRL algorithm is deep policy gradient, and the model consists of two hidden layers with 128 and 64 neurons, respectively, followed by ReLU activation functions. The output dimension is set to be the same as the number of actions, we set $J = 5$ in (9).

In the simulation scenario, we set $T_b = 1$ ms, which is the duration of an LTE subframe. We also set $T_s = 1$ s, or equivalently $H = 1000$, meaning the agent makes decisions every 1 second, therefore, the number of PRBs remains fixed for the subsequent 1000 time slots. We consider different QoS requirements for MVNOs in terms of maximum transmission latency. Two different services with maximum transmission latencies of 5 ms and 10 ms are considered. For the probabilistic constraint, we examine two different epsilon values: $\epsilon = 0.1$ and $\epsilon = 0.3$.

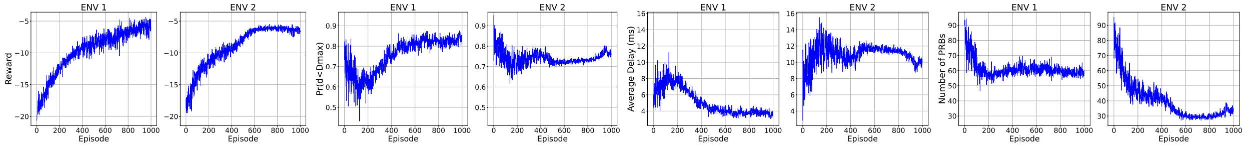
To simulate mobility and time-varying channel conditions, we introduce a Doppler frequency range of 5 Hz to 50 Hz. The number of users per slice ranges from 2 to 6. Additionally, for each user, we assume the data request packet sizes can follow three different distributions: small, medium, and large. Finally, we assume a total of 150 available PRBs in each cell.

Fig. 5 shows the convergence plots of the DRL agent during training in two different environments. In ENV1, we have $D^{max} = 5$ ms and $\epsilon = 0.1$, while in ENV2, we have $D^{max} = 10$ ms and $\epsilon = 0.3$. As we can see, the average delay goes below the D^{max} in both cases, while the satisfaction probability getting close to $1 - \epsilon$ as the model converges. Based on the wireless environment and QoS requirement, the agent is converging to the optimal number of PRBs, which for ENV1 is somewhere around 60 and for ENV2 is close to 30. It can be observed that the proposed reward function can appropriately balances the excessive use of PRBs and maximum delay violation while providing a smooth convergence.

In Figure 4, we compare our proposed solution PDA-DRL, with four other baselines: Mean Delay DRL (MD-DRL), Fixed-Av (Average Fixed), Fixed-Max (Maximum Fixed), and the heuristic policy, within a simulation scenario set at $D^{max} = 5$ ms and $\epsilon = 0.1$. For MD-DRL, the DRL model is trained with a reward function that maximizes when the average transmission delay is $5ms$. This criterion was selected to establish a DRL baseline and consider the average performance, as is common in many state-of-the-art solutions. The Fixed-Av policy employs a predetermined number of PRBs based on the average traffic demand of the slice, whereas the Fixed-Max policy uses a sufficient number of PRBs to ensure a 100% probability of meeting the delay constraint during peak traffic. This baseline assesses the excessive PRB usage required for an MVNO to perfectly satisfy QoS constraints with a fixed approach. Conversely, the heuristic policy dynamically adjusts the number of PRBs in response to the network's immediate state, increasing

Table 1: Performance comparison of different policies.

Policy	PDA-DRL	MD-DRL	Heuristic	Fixed-Av	Fixed-Max
Average PRB utilization	56.19	54.81	54.68	60.0	120.0
$Pr(d_q < D^{max})$	0.89	0.75	0.78	0.72	1.0
Mean delay (ms)	3.01	4.86	4.47	6.03	1.11
Delay STD (ms)	1.65	2.48	2.26	1.60	0.26
Average reward	-1.69	-4.39	-3.45	-5.01	-18.37



(a) Reward. (b) Probabilistic constraint. (c) Average delay. (d) Number of PRBs.

Figure 5: Convergence plots of different features in two distinct environments with different QoS requirements.

PRBs when constraint (6) is not met and reducing them otherwise, to avoid excess usage.

As observed in Fig. 4, the fixed policies struggle to dynamically adapt to fluctuating traffic, leading to a trade-off between excessive PRB usage and QoS degradation during varying traffic demands. While Fixed-Max ensures a 100% guarantee of transmission delay staying below 5 ms, it consumes more than twice the number of PRBs compared to the other baselines. The MD-DRL, as expected, maintains an average delay below 5 ms, but the delay standard deviation (STD) indicates that still a considerable portion of packets experience transmission delay longer than 5 ms. The heuristic policy, which adjusts PRBs based on current satisfaction probabilities, tends to fluctuate between satisfied and unsatisfied QoS states. Although it performs better than the Fixed baselines and MD-DRL, it still falls more than 12% short of the QoS constraint. On the other hand, the PDA-DRL uses almost the same number of PRBs as the heuristic and MD-DRL approaches while closely meeting the QoS constraint within a 1% margin. **Notably, the PDA-DRL approach demonstrates superior performance by achieving a 38% reduction in average delay and 33% reduction in STD compared to MD-DRL. This illustrates that a percentile-based formulation can also provide a tighter bound on average delay.** The summarized performance metrics of these methods are reported in Table 1.

In Fig. 6, we present a comparison of various personalization methods' performance across 10 distinct environments. These environments are characterized by significant differences in QoS requirements (ϵ, D^{max}) and wireless conditions, including average CQI, Doppler frequency, number of users, and types of traffic. As expected, methods like FedAV, which rely on naively averaging model weights, show a poor performance. Similarly, aggregation based on model weight and feature similarity does not significantly enhance performance. In contrast, reward-based personalization demonstrates superior performance, outstripping all other aggregation methods by a considerable margin and rivaling the performance of local models. It is worth noting that, to facilitate comparison in Fig. 6, we introduced a positive bias to the obtained rewards, as they are inherently negative. For the aggregation coefficient in (16), we set $\beta = 3$ and $T = 10$.

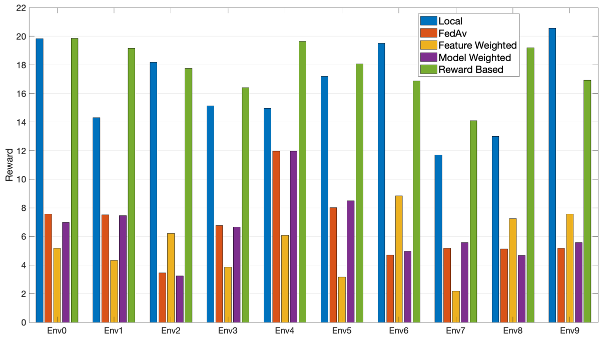


Figure 6: Comparison between different aggregation methods.

In Appendix C, we have validated our DRL solution through experimental framework which is built on top of Colosseum (Bonati et al., 2021c) and Scope(Bonati et al., 2021a).

9. Conclusion

We proposed PDA-DRL for delay-aware RAN slicing in O-RAN, optimizing PRB allocation via LLN-based rewards and reward shaping. Our solution outperforms baselines (e.g., MD-DRL) and introduces collaborative DRL personalization, excelling in diverse QoS scenarios.

References

Abouaomar, A., Taik, A., Filali, A., and Cherkaoui, S. Federated deep reinforcement learning for open ran slicing in 6g networks. *IEEE Communications Magazine*, 2022.

Akman, A., Oliver, P., Jones, M., Tehrani, P., Hoffmann, M., and Li, J. Energy saving and traffic steering use case and testing by o-ran ric xapp/rapp multi-vendor interoperability. In *2024 IEEE 100th Vehicular Technology Conference (VTC2024-Fall)*, pp. 1–6. IEEE, 2024.

Bakri, S., Frangoudis, P. A., Ksentini, A., and Bouaziz, M. Data-driven ran slicing mechanisms for 5g and beyond. *IEEE Transactions on Network and Service Management*, 18(4):4654–4668, 2021.

Bonati, L., D'Oro, S., Basagni, S., and Melodia, T. Scope: An open and softwarized prototyping platform for nextg systems. In *Proceedings of the 19th Annual International Conference on Mobile Systems, Applications, and Services*, pp. 415–426, 2021a.

Bonati, L., D'Oro, S., Polese, M., Basagni, S., and Melodia, T. Intelligence and learning in o-ran for data-driven nextg cellular networks. *IEEE Communications Magazine*, 59 (10):21–27, 2021b.

Bonati, L., Johari, P., Polese, M., D'Oro, S., Mohanti, S., Tehrani-Moayyed, M., Villa, D., Shrivastava, S., Tassie, C., Yoder, K., et al. Colosseum: Large-scale wireless experimentation through hardware-in-the-loop network emulation. In *2021 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, pp. 105–113. IEEE, 2021c.

Cao, Y., Lien, S.-Y., Liang, Y.-C., Chen, K.-C., and Shen, X. User access control in open radio access networks: A federated deep reinforcement learning approach. *IEEE Transactions on Wireless Communications*, 21(6):3721–3736, 2021.

D'Oro, S., Bonati, L., Polese, M., and Melodia, T. Orchestran: Network automation through orchestrated intelligence in the open ran. In *IEEE INFOCOM 2022-IEEE Conference on Computer Communications*, pp. 270–279. IEEE, 2022.

Elayoubi, S. E., Jemaa, S. B., Altman, Z., and Galindo-Serrano, A. 5g ran slicing for verticals: Enablers and challenges. *IEEE Communications Magazine*, 57(1):28–34, 2019.

Ericsson. Radio access network (ran) digital twins. White paper, 2023.

Filali, A., Mlika, Z., Cherkaoui, S., and Kobbane, A. Dynamic sdn-based radio access network slicing with deep reinforcement learning for urllc and embb services. *IEEE Transactions on Network Science and Engineering*, 9(4): 2174–2187, 2022.

Ge, C., Xia, S., Chen, Q., and Adachi, F. Learning based on graph: A joint interference coordination for cluster-wise distributed mu-mimo. *IEEE Communications Letters*, 2023.

Hua, Y., Li, R., Zhao, Z., Chen, X., and Zhang, H. Gan-powered deep distributional reinforcement learning for resource management in network slicing. *IEEE Journal on Selected Areas in Communications*, 38(2):334–349, 2019.

Huang, Y., Chu, L., Zhou, Z., Wang, L., Liu, J., Pei, J., and Zhang, Y. Personalized cross-silo federated learning on non-iid data. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021.

Joda, R., Pamuklu, T., Iturria-Rivera, P. E., and Erol-Kantarci, M. Deep reinforcement learning-based joint user association and cu-du placement in o-ran. *IEEE Transactions on Network and Service Management*, 2022.

Kalntis, M. and Iosifidis, G. Energy-aware scheduling of virtualized base stations in o-ran with online learning. In *GLOBECOM 2022-2022 IEEE Global Communications Conference*, pp. 6048–6054. IEEE, 2022.

Karapantelakis, A., Alizadeh, P., Alabassi, A., Dey, K., and Nikou, A. Generative ai in mobile networks: a survey. *Annals of Telecommunications*, pp. 1–19, 2023.

Kougioumtzidis, G., Vlahov, A., Poulkov, V. K., Lazaridis, P. I., and Zaharis, Z. D. Deep learning-aided qoe prediction for virtual reality applications over open radio access networks. *IEEE Access*, 2023.

Lacava, A., Polese, M., Sivaraj, R., Soundrarajan, R., Bhati, B. S., Singh, T., Zugno, T., Cuomo, F., and Melodia, T. Programmable and customized intelligence for traffic steering in 5g networks using open ran architectures. *arXiv preprint arXiv:2209.14171*, 2022.

Lien, S.-Y., Deng, D.-J., and Chang, B.-C. Session management for urllc in 5g open radio access network: A machine learning approach. In *2021 International Wireless Communications and Mobile Computing (IWCMC)*, pp. 2050–2055. IEEE, 2021.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. Communication-efficient learning of deep networks from decentralized data. In *Artificial intelligence and statistics*, pp. 1273–1282. PMLR, 2017.

Mei, J., Wang, X., Zheng, K., Boudreau, G., Sediq, A. B., and Abou-Zeid, H. Intelligent radio access network slicing for service provisioning in 6g: A hierarchical deep reinforcement learning approach. *IEEE Transactions on Communications*, 69(9):6063–6078, 2021.

Messaoud, S., Bradai, A., Ahmed, O. B., Quang, P. T. A., Atri, M., and Hossain, M. S. Deep federated q-learning-based network slicing for industrial iot. *IEEE Transactions on Industrial Informatics*, 17(8):5572–5582, 2020.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M., Fidjeland, A. K., Ostrovski, G., et al. Human-level control through deep reinforcement learning. *nature*, 518(7540): 529–533, 2015.

Motalleb, M. K., Shah-Mansouri, V., Parsaeefard, S., and López, O. L. A. Resource allocation in an open ran system using network slicing. *IEEE Transactions on Network and Service Management*, 20(1):471–485, 2022.

Nagib, A. M., Abou-zeid, H., and Hassanein, H. S. Accelerating reinforcement learning via predictive policy transfer in 6g ran slicing. *IEEE Transactions on Network and Service Management*, 2023.

Niknam, S., Roy, A., Dhillon, H. S., Singh, S., Banerji, R., Reed, J. H., Saxena, N., and Yoon, S. Intelligent o-ran for beyond 5g and 6g wireless networks. In *2022 IEEE Globecom Workshops (GC Wkshps)*, pp. 215–220. IEEE, 2022.

O-RAN Alliance. O-RAN Architecture Description 10.0. Technical Specification (TS) O-RAN.WG1.OAD-R003-v10.00, O-RAN Alliance, Oct 2023. R003. [Online]. Available: https://www.o-ran.org/specifications.

Orhan, O., Swamy, V. N., Tetzlaff, T., Nassar, M., Nikopour, H., and Talwar, S. Connection management xapp for o-ran ric: A graph neural network and reinforcement learning approach. In *2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pp. 936–941. IEEE, 2021.

Polese, M., Bonati, L., D'Oro, S., Basagni, S., and Melodia, T. Understanding o-ran: Architecture, interfaces, algorithms, security, and research challenges. *IEEE Communications Surveys & Tutorials*, 2023.

Popovski, P., Trillingsgaard, K. F., Simeone, O., and Durisi, G. 5g wireless network slicing for embb, urllc, and mmtc: A communication-theoretic view. *Ieee Access*, 6:55765–55779, 2018.

Raeis, M., Tizghadam, A., and Leon-Garcia, A. Queue-learning: A reinforcement learning approach for providing quality of service. *arXiv preprint arXiv:2101.04627*, 2021.

Reus-Muns, G., Upadhyaya, P. S., Demir, U., Stephenson, N., Soltani, N., Shah, V. K., and Chowdhury, K. R. Senseoran: O-ran based radar detection in the cbrs band. *IEEE Journal on Selected Areas in Communications*, 2023.

Rezazadeh, F., Zanzi, L., Devoti, F., Chergui, H., Costa-Pérez, X., and Verikoukis, C. On the specialization of fdrl agents for scalable and distributed 6g ran slicing orchestration. *IEEE Transactions on Vehicular Technology*, 2022.

Setayesh, M., Bahrami, S., and Wong, V. W. Resource slicing for embb and urllc services in radio access network using hierarchical deep learning. *IEEE Transactions on Wireless Communications*, 21(11):8950–8966, 2022.

Sun, C., Pawar, U., Khoja, M., Foukas, X., Marina, M. K., and Radunovic, B. Spotlight: Accurate, explainable and efficient anomaly detection for open ran. In *Proceedings of the 30th Annual International Conference on Mobile Computing and Networking*, pp. 923–937, 2024a.

Sun, K., Tao, C., Johari, P., D'Oro, S., Polese, M., Bonati, L., Melodia, T., Rajendran, G., Kundu, N., Chakfeh, Y., Raghothaman, B., D'angelo, M., Agarwal, R., Kundu, L., Lin, X., and Dick, C. Research report on digital twin ran use cases, May 2024b. Report ID: RR-2024-07.

Tehrani, P., Restuccia, F., and Levorato, M. Federated deep reinforcement learning for the distributed control of nextg wireless networks. In *2021 IEEE International Symposium on Dynamic Spectrum Access Networks (DySPAN)*, pp. 248–253. IEEE, 2021.

Vannella, F., Jeong, J., and Proutiere, A. Off-policy learning in contextual bandits for remote electrical tilt optimization. *IEEE Transactions on Vehicular Technology*, 72(1): 546–556, 2022.

Wijethilaka, S. and Liyanage, M. Survey on network slicing for internet of things realization in 5g networks. *IEEE Communications Surveys & Tutorials*, 23(2):957–994, 2021.

Wu, W., Chen, N., Zhou, C., Li, M., Shen, X., Zhuang, W., and Li, X. Dynamic ran slicing for service-oriented vehicular networks via constrained learning. *IEEE Journal on Selected Areas in Communications*, 39(7):2076–2089, 2020.

Xavier, B. M., Dzaferagic, M., Collins, D., Comarela, G., Martinello, M., and Ruffini, M. Machine learning-based early attack detection using open ran intelligent controller. *arXiv preprint arXiv:2302.01864*, 2023.

Yang, P., Xi, X., Quek, T. Q., Chen, J., Cao, X., and Wu, D. Ran slicing for massive iot and bursty urllc service multiplexing: Analysis and optimization. *IEEE Internet of Things Journal*, 8(18):14258–14275, 2021.

A. Reward Function Design

To devise a suitable reward function, it is essential to consider the progress of the υth MVNO, acting as a DRL agent, towards achieving the objectives of the sequential optimization problem (5), while also ensuring compliance with the constraint (6). Consequently, the reward function should focuses on tracking packets that meet the transmission delay deadline and imposes penalties for excessive PRB utilization. Following a similar approach to that used in (Raeis et al., 2021) for RL reward design, for each packet $f_{j,k,q}$ we define variable z_q as follows:

$$z_q = \begin{cases} 1 & \text{if } \sum_{t=t_0}^{t_0+D^{max}} C_{j,k}^t l_{j,k,q}^t \geq f_{j,k,q}^{t_0} \\ 0 & \qquad O.W. \end{cases}$$

Since our focus here is on a single DRL agent, we will drop the index υ. For the sake of simplicity in notation, we will use the index q to refer to a packet for the remainder of the analysis.

Now we can define an immediate reward, based on the delay requirement of packet q as:

$$r_q = \begin{cases} u_o & \text{if } z_q = 0 \\ u_1 & \text{if } z_q = 1 \end{cases}$$

$$r_t = r(s_t, \pi_\theta(s_t)) = \sum_{q \in \mathcal{A}_{T_s}} \frac{r_q}{\mathbb{E}[n_a]} - N_{T_s} \tag{17}$$

where $\mathcal{A}_{T_s}$ is the set of arrivals in time slot T_s and $\mathbb{E}[n_a]$ is the average number of arrivals.

Now, the average reward per time step can be calculated using the defined reward function as follows:

$$J_{\pi_\theta} = \int_{\mathcal{S}} p_{\pi_\theta}(s) \mathbb{E}\big\{r(s, \pi_\theta(s))\big\} d_s \tag{18}$$

where $p_{\pi_\theta}(s)$ is the steady state distribution of the states when following policy π_θ and is defined as follows:

$$p_{\pi_\theta}(s') = \int_{\mathcal{S}} \sum_{t=1}^{\infty} \gamma^{t-1} p(s) p\big(s'|s, t, \pi_\theta\big) \tag{19}$$

In the above equation $p\big(s'|s, t, \pi_\theta\big)$ denotes the transition probability from state s to state s' after t time step under policy π_θ.

Even with a fixed state s and action $a = \pi_\theta(s)$, the reward $r(s, a)$ can still be random due to the stochasticity arising from random packet arrivals and packet sizes in the subsequent slicing time slot. To calculate this expectation, we would have:

$$\mathbb{E}\big\{r(s_t, \pi_\theta(s_t))|s_t = s\big\} = \mathbb{E}\big\{ \sum_{q \in \mathcal{A}_{T_s}} \frac{r_q}{\mathbb{E}\big\{n_a\big\}} - N_{T_s}|s_t = s\big\}$$

defining $\mathcal{Z}$ as the set of packets in the current time slot which meet the deadline requirements:

$$\mathcal{Z} = \big\{z|z_q = 1, \quad \forall q\big\} \tag{20}$$

then the expectation would be:

$$\begin{aligned} &\mathbb{E}\big\{ \sum_{i \in \mathcal{Z}} \frac{u_1}{\mathbb{E}\big\{n_a\big\}} + \sum_{i \in \mathcal{Z}^{\rfloor}} \frac{u_0}{\mathbb{E}\big\{n_a\big\}} - N_{T_s}|s\big\} \\ &= u_1 \frac{\mathbb{E}\big\{|\mathcal{Z}|\big|s\big\}}{\mathbb{E}\big\{n_a\big\}} + u_0 \frac{\mathbb{E}\big\{|\mathcal{Z}^{\rfloor}|\big|s\big\}}{\mathbb{E}\big\{n_a\big\}} - \mathbb{E}\big\{N_{T_s}|s\big\} \end{aligned}$$

Assuming $T_s >> T_b$, then using LLN we can approximate the previous term as:

$$\approx u_1 Pr(d_q < D^{max}|s) + u_0 Pr(d_q > D^{max}|s) - \mathbb{E}\{N_{T_s}|s\}$$

By choosing $u_1 = \lambda\epsilon$ and $u_0 = -\lambda(1-\epsilon)$ we would have:

$$J_{\pi_\theta} = \lambda(Pr(d_q < D^{max}) - (1-\epsilon)) - \mathbb{E}\{N_{T_s}\} \tag{21}$$

where λ specifies the trade-off between the QoS constraint and the utilized PRBs. Interestingly with this rewards design procedure, the value of J_{π_θ} is similar to the dual form of the main optimization problem. If we consider the minimization problem (5) and the constraint (6), the associated Lagrangian function would be:

$$L_{\theta,\lambda} = -\mathbb{E}\{N_{T_s}\} + \lambda(Pr(d_q < D^{max}) - (1-\epsilon)) \tag{22}$$

We have thus proven that maximizing the average reward J_{π_θ} with respect to θ is equivalent to computing the Lagrangian dual function associated with problem (5).

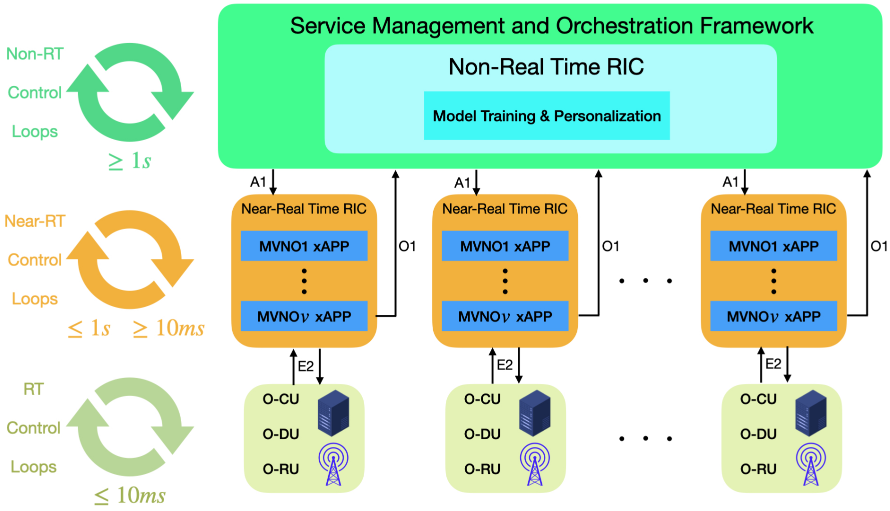


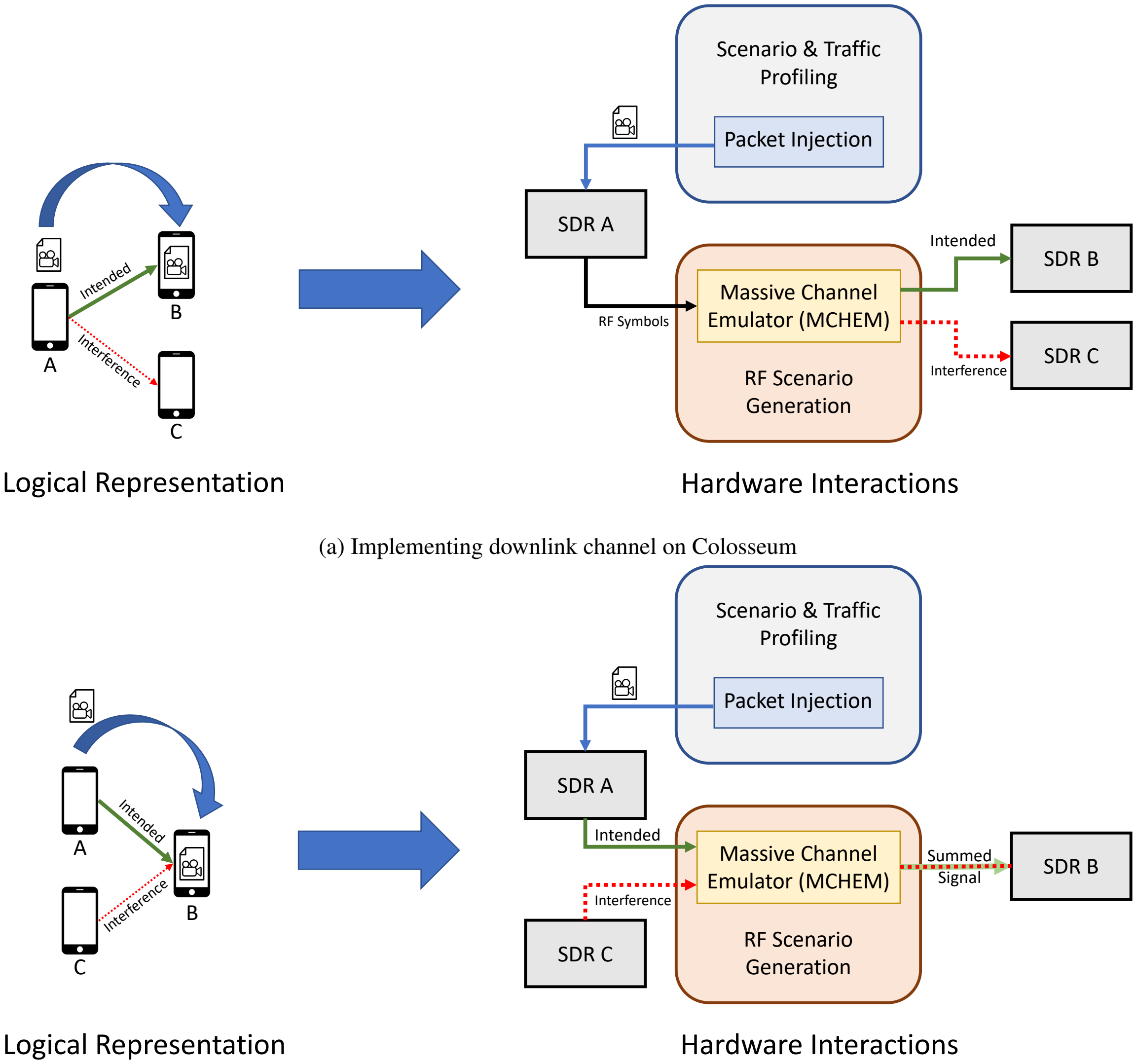
Figure 7: ORAN compliant system architecture.

B. ORAN Compliance

In Fig. 7, we depict the design we've crafted, which aligns with the principles of the O-RAN initiative (O-RAN Alliance, 2023). O-RAN leverages cloud RAN (C-RAN) principles and the increasingly software-defined implementations of wireless communications and networking functions. Unlike legacy interfaces that are vendor-specific and controlled by major industry players, it defines open interfaces and an open architecture to foster innovation at all layers. Central to O-RAN's architecture are the non-real-time (non-RT) RIC and the near-real-time (near-RT) RIC. The non-RT RIC is responsible for RAN optimization over broader timescales, such as seconds to minutes or longer periods of hours or days, and typically involves the training of ML models and the formulation of control policies. The model personalization discussed in Sec 7 would occur in this layer, as it could have access to all DRL models, and model aggregation would take place on a larger timescale, like days.

The resulting models are then communicated through the A1 interface to the near-RT RICs distributed across the network. The near-RT RIC facilitates optimization and control functions within a shorter timescale, from 10 milliseconds to 1 second, and oversees monitoring of both the O-RAN Central Unit (O-CU) and the O-RAN Distributed Unit (O-DU), which host eNBs/gNBs as virtualized functions. As shown in Fig. 7, each MVNO can run its own DRL-based slicing xApps in the near-RT RIC, tailored for that specific RAN region and the MVNO's QoS requirements.

Local DRL agents, or xApps, associated with the near-RT RICs, would collect data, make local decisions, and leverage the E2 interface to communicate these radio policies to the RAN. The E2 interface also serves as the conduit for transmitting RAN's KPIs back to the agents, for example, the state of the network, which includes the PRB utilization, buffer size, channel CQI, and related network features needed for ongoing model training and monitoring. This bidirectional communication facilitates a continuous feedback loop, enabling real-time adaptation and optimization of network operations.



(a) Implementing downlink channel on Colosseum

(b) Implementing uplink channel on Colosseum

Figure 8: Demonstration of signal transmission in Colosseum.

C. Experimental Framework

In this section we describe our experimental setup which utilizes software-defined radios (SDR) in order to validate our proposed DRL model results under real-life scenarios. All of the experiments were performed on the world's largest RF emulator, Colosseum (Bonati et al., 2021c), which is an extensive network of high-end servers and SDR devices. The SDR hardware used in the Colosseum is USRP X310, a high-end SDR device with a 200 MHz bandwidth per channel. The Colosseum testbed is composed of 256 SDRs connected to another 256 SDRs which enables it to output up to 65K independent RF channels. Although the testbed is built for wireless experiments, all SDRs are connected via coaxial cables. To emulate the wireless environment, every SDR-generated signal goes through a Massive Channel Emulator (MCHEM) which is composed of 512 complex-valued FIR taps that are used to apply environmental artifacts to the signals such as fading, path-loss, and interference. For example as shown in Fig. 8a, when SDR A sends a signal towards SDR B, SDR C would receive a distorted copy of the signal as interference based on the used RF scenario (e.g. applying node mobility or multi-path effects). Similarly in Figure 8b, when SDR A sends a signal to SDR B while having SDR C is transmitting in the background, SDR B would receive the summed signal of SDR A and SDR B transmissions. This testbed design would provide a tighter control over the RF environment that is being tested instead of radiating the RF signal into the air.

The design of our experiment, as shown in Fig. 9, starts with srsRAN (formerly known as srsLTE), which is an open-source implementation of the LTE and 5G standards. The srsRAN is then run on top of the SDR devices and configured as either base stations or mobile devices, depending on the experimental scenario. To segment the srsRAN radio resources for slicing operations, we utilize SCOPE (Bonati et al., 2021a), a physical-layer slicing framework that groups the PRBs of the LTE network into separate slices. According to the LTE standard, a PRB consists of 12 subcarriers, each with a bandwidth of 15 kHz, and a duration of 0.5 ms. Hence, a single PRB has a total bandwidth of 180 kHz in frequency and a duration of 0.5 ms. By employing the SCOPE framework, we can dynamically allocate a dedicated number of PRBs to individual network slices. On top of SCOPE, we implement a custom-made slice controller that serves higher-layer applications, such as a DRL agent.

The slice controller has three primary functions: dynamically allocating the PRBs to each slice, controlling the amount of injected traffic into the LTE network based on the experimental scenario, and producing datasets at runtime for each mobile device. When invoked, the slice controller spawns eNodeBs (LTE RAN) and UEs (User Equipment). Each UE attaches itself to its assigned eNodeB via the LTE network using the SDR interface. Then each eNodeB initiates the network slices with its assigned UEs based on the initial experiment configurations. Each slice has a dedicated slice controller, which can be controlled by a dedicated DRL agent representing the MVNO discussed in previous sections of the paper. Once a UE connects to its associated eNodeB, the UE establishes an additional connection with the eNodeB through Colosseum's internal network. This additional side channel is used to report network statistics, such as delay as shown in Fig. 2, back to the slice controller without utilizing LTE resources used in the experiment. In other words, for each packet sent by the eNodeB, the UE sends packet statistics back to its eNodeB via the Colosseum side channel.

These packet delivery statistics are used to generate a new system state, which is then reported back to the DRL agent every 250ms. The chosen system resolution of 250ms is limited by srsRAN, which updates its internal state four times per second. At each new system state, the DRL agent sends network traffic updates and PRB allocation requests to the slicing tool. The slicing tool updates the injection parameters of the traffic profile and reallocates the PRBs among the slices based on the agent's requests. At the end of each system state, SCOPE outputs the statistics of that state. The slice controller merges its own statistics with SCOPE's, such as the total number of PRBs used in the last 250ms. Fig. 9 provides a summary of this procedure.

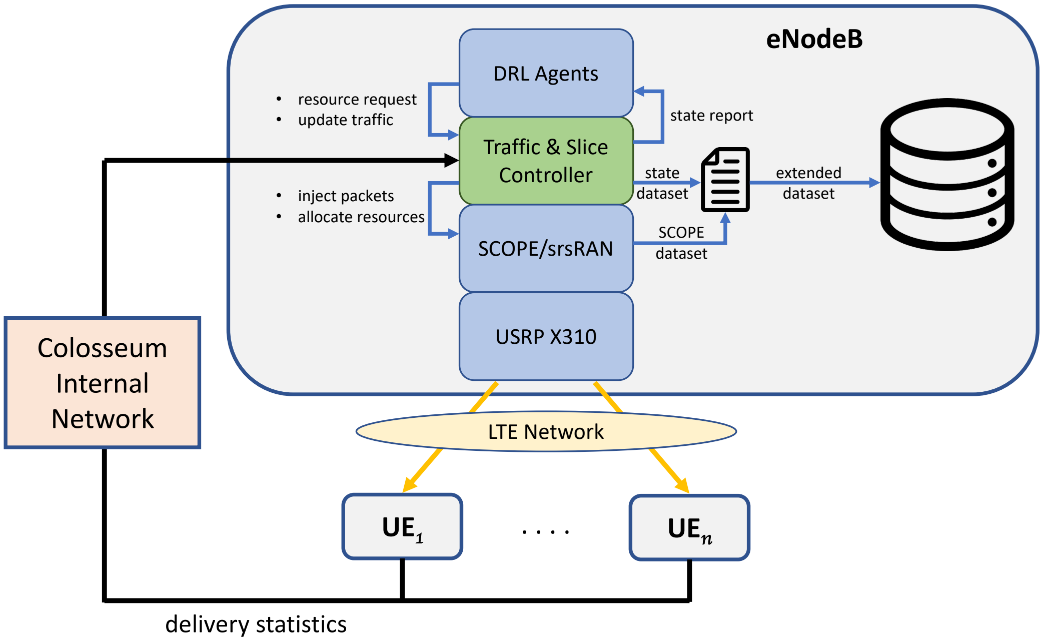
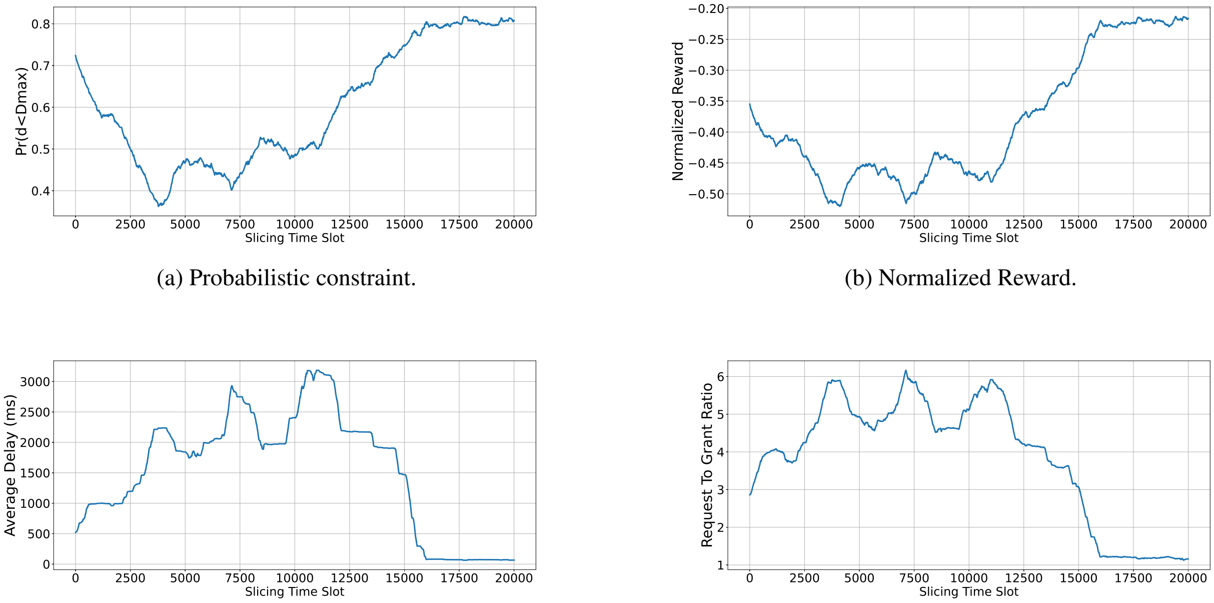


Figure 9: Experimental framework architecture.



(a) Probabilistic constraint.

(b) Normalized Reward.

(c) Average delay.

(d) Requested-to-granted PRBs ratio.

Figure 10: Convergence plots in experimental setting.

C.1. Experimental Results

Here we describe the experimental results. In the Colosseum, we used 50 available PRBs with increments/decrements of 1 resource block group (3 PRBs), setting our action set to $-9, -6, -3, 0, 3, 6, 9$. Due to srsRAN limitations, the agent controls the slice and computes network statistics every 250 ms. To improve sample efficiency, we employed Deep Q-learning with a learning rate of $1e-4$, a decay factor of 0.9, a batch size of 64, and an epsilon-greedy exploration strategy starting at 0.5 and decreasing to 0.05.

Figure 10 shows the convergence curves during online training over 20,000 time slots, with a moving average window of 5000. The agent learns to manage PRBs to reduce delay effectively. The requested-to-granted PRBs ratio, ideally balanced around 1 to prevent excessive demand and resource wastage, converges towards this ideal by the end of training. Additionally, the average delay decreases, and the probability of satisfaction meets the constraint ($\epsilon = 0.2$).