FAIR-Q: Fairness and Adaptive Intelligent Resource Management with QoS Optimization in Dynamic 6G Radio Access Networks

Ioan-Sorin Comșa[1], Per Bergamin[1], Gabriel-Miro Muntean[2], Purav Shah[3], and Ramona Trestian[3]
[1]Institute for Research in Open-, Distance- and eLearning,
Swiss Distance University of Applied Sciences, CH-3900, Brig, Switzerland,
[2]School of Electronic Engineering, Dublin City University, D09 V209, Dublin, Ireland,
[3]London Digital Twin Research Centre, Middlesex University London, NW4 4BT, Hendon, London, U.K.
E-mails: {ioan-sorin.comsa, per.bergamin}@ffhs.ch, gabriel.muntean@dcu.ie, {p.shah, r.trestian}@mdx.ac.uk

***Abstract*—The advent of 6G networks brings diverse services, such as immersive multimedia, augmented reality, and massive IoT, each with stringent requirements for Quality of Service (QoS) and fairness. These challenges expose the limitations of traditional scheduling algorithms, which struggle to dynamically adapt to evolving network conditions. To address this, we propose FAIR-Q, a novel Fairness and Adaptive Intelligent Resource Management framework with QoS optimization driven by Reinforcement Learning (RL) approach. FAIR-Q integrates a multi-objective reward function to optimize fairness, packet loss, delay, and rate constraints. The framework features two key controllers: a Parameterization Controller, which dynamically adjusts scheduling parameters to ensure fairness, and a Scheduling Rule Controller, which intelligently selects scheduling rules to adapt to real-time network conditions and align with QoS requirements. Simulation results demonstrate up to a 15% improvement in fairness and QoS satisfaction compared to static scheduling methods, underscoring the adaptability and scalability of FAIR-Q in dynamic 6G Radio Access Networks.**

***Index Terms*—Multi-objective Optimization, Reinforcement Learning, Scheduling, QoS, Fairness, Dynamic Networks.**

I. Introduction

The emergence of 6G networks promises to revolutionize multimedia services such as augmented and virtual reality, holographic communications, and ultra-high-definition streaming, all of which impose stringent Quality of Service (QoS) requirements, including ultra-low latency, high reliability, and adaptive data rates, and fairness among users [1]. These demands place significant strain on traditional Radio Access Networks (RANs). Open RAN, with its modular and flexible architecture, addresses these challenges by decoupling hardware from software, enabling dynamic resource management, programmable interfaces, and integration of advanced algorithms to adapt to evolving network conditions [2].

Reinforcement Learning (RL) has emerged as a promising approach to Radio Resource Management (RRM) within the Open RAN ecosystem. By learning optimal actions through interaction with the environment, RL dynamically adapts to changing network conditions, improving scheduling, resource allocation, and interference management [3]. Its ability to handle high-dimensional state spaces, adapt to varying traffic patterns, and enable real-time decision-making makes RL an ideal solution for addressing challenges in 6G RRM [4].

This paper introduces FAIR-Q, a novel RL-based framework designed to simultaneously address fairness and QoS objectives in scheduling diverse data traffic. FAIR-Q employs a multi-objective optimization strategy utilizing two specialized controllers: a Parameterization Controller, which dynamically adjusts scheduling rule parameters to meet fairness requirements, and a Rule Selector Controller, which intelligently selects the appropriate scheduling rule to achieve multiple objectives, including user fairness, delay, Packet Loss Ratio (PLR), and rate constraints. This intelligent and adaptive approach ensures enhanced resource management and scalability in evolving network environments.

II. Related Work

Early directions in applying RL in RRM scheduling were proposed in [5] and [6], where a Q-learning framework was employed to dynamically select the scheduling rule at each Transmission Time Interval (TTI), enhancing throughput and fairness by learning optimal policies. Throughput-fairness tradeoff is optimized in [7] using deep RL for dynamic resource and user scheduling. In [8] and [9] multiple RL algorithms were trained to adapt the Generalized Proportional Fair (GPF) scheduling rule, specifically targeting the fairness requirements set by the Next Generation Mobile Networks (NGMN). Extending this line of research, in [10] authors conducted a comparative study of various RL algorithms, analyzing their effectiveness under different averaging strategies used for computing user throughput, with a focus on NGMN fairness objective. Also, throughput optimization is addressed in [11] using multi-agent RL with a graph attention network to allocate radio resources effectively in 6G sub-networks.

A deep RL method is proposed in [12] to enhance throughput and reduce delay in Open RAN, optimizing resource allocation for both enhanced Mobile Broadband (eMBB) and Ultra-Reliable Low-Latency Communication (URLLC) services. In [13] a deep transfer RL framework is proposed to optimize joint radio and cache resource allocation in 5G RAN slicing, achieving significant improvements in throughput and latency for eMBB and URLLC services. By focusing on delay and drop rate minimization in applications with stringent requirements, a novel scheduling framework is studied in [14],

leveraging RL approaches to dynamically select scheduling rules based on real-time network conditions. This multi-objective optimization through RL is further explored in [15], addressing diverse traffic demands by meeting strict rate, PLR, and delay requirements. This work is extended to heterogeneous traffic scenarios, demonstrating significant improvements in terms of PLR, throughput, and delay [16].

Building on prior work, this paper takes a comprehensive approach by integrating both fairness and diverse QoS requirements—such as rate, packet loss, and delay—into a unified RL-based scheduling framework. This holistic method ensures equitable resource allocation while simultaneously optimizing user satisfaction and network performance. The key benefits of this approach include improved fairness across users, enhanced adaptability to heterogeneous traffic conditions, and the ability to meet stringent QoS demands, making it well-suited for next-generation wireless networks.

III. System Model

The proposed FAIR-Q system model from Fig. 1 consists of two key components: the scheduling and resource allocation module, and the proposed RL framework. Scheduling prioritizes user requests during each TTI t, while resource allocation assigns radio resources to these prioritized users. As highlighted in the related work, the performance of scheduling (measured in terms of QoS provisioning and fairness assurance) relies heavily on the choice of the scheduling rule, especially in dynamic network and traffic conditions. The proposed RL framework is designed to dynamically select the appropriate scheduling rule at each TTI t, based on the instantaneous scheduler state $\mathbf{s}(t) \in \mathcal{S}$, which includes QoS indicators, traffic characteristics, and channel conditions. The scheduler state must be processed at each TTI to enable efficient RL operation, ensuring compliance with stringent QoS and fairness requirements under dynamic network conditions. To achieve the multi-objective goals of fairness and QoS, the proposed RL framework employs two specialized controllers: a Parameterization Controller, which tunes the parameters of the scheduling rule, and a Rule Selector, which determines the most suitable scheduling rule to apply at each TTI.

A. Problem Formulation

We consider downlink multimedia communications using OFDMA radio access scheme, where the available bandwidth is divided into J equally sized Resource Blocks (RBs). Let $\mathcal{J} = \{1, 2, \ldots, J\}$ represent the set of RBs to be shared among the active users, denoted by $\mathcal{I}_t = \{1, 2, \ldots, I_t\}$, where I_t is the number of active users at TTI t. The scheduler is tasked with allocating each RB $j \in \mathcal{J}$ to user $i \in \mathcal{I}_t$ in each TTI, aiming to maximize the long-term fulfillment of QoS and fairness requirements. To formalize the objectives, let $\mathcal{O} = \{1, 2, \ldots, O\}$ denote the set of objectives targeted by the scheduling process within RRM. In this model, we consider $O = 4$ objectives, where $o = 1$ addresses NGMN fairness, followed by objectives for rate ($o = 2$), delay ($o = 3$), and PLR ($o = 4$). For each user $i \in \mathcal{I}_t$, the objective $o \in \mathcal{O}$ is addressed by employing a specific function $\mathcal{F}_o$ which dictates the scheduling of users with the highest deficit

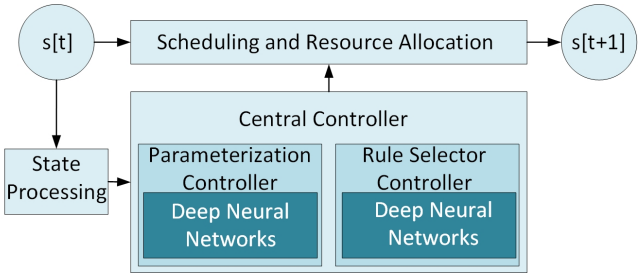

Fig. 1: Proposed FAIR-Q System Model

concerning the selected objective o. Given the diverse QoS requirements, dynamic network conditions, user preferences, and traffic characteristics, a variety of scheduling rules have been developed to address individual or combined objectives. Let $\mathcal{D} = \{1, 2, \ldots, D\}$ denote the set of scheduling rules. At each TTI, the scheduling entity must determine the objective to address, the rule to apply, the users to prioritize, and the RBs to allocate in order to meet the QoS and fairness requirements. This task forms a multi-objective combinatorial scheduling problem, mathematically expressed as follows:

$$\max_{x,y,z} \sum_{o} \sum_{d} \sum_{i} \sum_{j} x_{o,d}(t) \cdot y_{d,i}(t) \cdot z_{i,j}(t) \cdot \mathcal{F}_o\left(\mathcal{Q}_{o,i}, \frac{\lambda_{i,j}}{\Lambda_i}\right) \cdot \lambda_{i,j}, \tag{1}$$

$s.t.$

$$q_{o,i} \; constrained \; by \; \bar{q}_{o,i}, \quad \forall o \in \mathcal{O}, \; \forall i \in \mathcal{I}_t \tag{1.a}$$

$$x_{o,d}(t) \in \{0, 1\}, \quad \forall o \in \mathcal{O}, \forall d \in \mathcal{D}, \tag{1.b}$$

$$y_{d,i}(t) \in \{0, 1\}, \quad \forall d \in \mathcal{D}, \forall i \in \mathcal{I}_t, \tag{1.c}$$

$$z_{i,j}(t) \in \{0, 1\}, \quad \forall i \in \mathcal{I}_t, \forall j \in \mathcal{J}, \tag{1.d}$$

$$\sum\nolimits_{o} x_{o,d}(t) \geq 1, \quad d = 1, 2, ..., D, \tag{1.e}$$

$$\sum\nolimits_{d} y_{d,i}(t) = 1, \quad i = 1, 2, ..., I_t, \tag{1.f}$$

$$\sum\nolimits_{i} z_{i,j}(t) \leq 1, \quad j = 1, 2, ..., J. \tag{1.g}$$

In (1), $\lambda_{i,j}$ represents the achievable rate on RB $j \in \mathcal{J}$ for user $i \in \mathcal{I}_t$, emphasizing that the primary objective of the scheduling process is to maximize system throughput by exploiting multi-user frequency diversity. This rate maximization objective is modulated by the scheduling rule $\mathcal{F}_o$, which prioritizes fairness by incorporating the ratio $\lambda_{i,j}/\Lambda_i$, where Λ_i is the average throughput of user $i \in \mathcal{I}_t$. This ratio ensures that users with relatively higher average throughput are deprioritized in favor of those with lower average throughput or potentially more unfavorable channel conditions. The NGMN fairness requirement provides a framework for balancing fairness with throughput maximization. Beyond fairness, the rate maximization objective is further refined by the QoS dataset $\mathcal{Q}_{o,i} = \{q_{o,i}, \bar{q}_{o,i}\}$, where $q_{o,i}$ represents the QoS indicators (e.g., throughput, delay, PLR) for objective $o \in \mathcal{O}$, and $\bar{q}_{o,i}$ denotes the associated QoS requirement. A specific QoS objective $o \in \mathcal{O}$ is fulfilled for user i when $q_{o,i}$ meets $\bar{q}_{o,i}$. In such cases, the scheduling process prioritizes users with the lowest degree of QoS satisfaction for the given objective $o \in \mathcal{O}$. Within the constraints of fairness and QoS as outlined in (1.a), the combinatorial optimization problem seeks to determine the best binary decision variables defined

in (1.b), (1.c), and (1.d). These decision variables include:

- Objective and rule selection variable: $x_{o,d}[t]$, which indicates whether objective $o \in \mathcal{O}$ is selected and scheduling rule $d \in \mathcal{D}$ is applied. If $x_{o,d}(t) = 1$, objective o is targeted using rule d; otherwise, $x_{o,d}(t) = 0$. Constraints (1.e) ensure that at least one objective is associated with each scheduling rule $d \in \mathcal{D}$.
- Rule assignment variable: $y_{d,i}[t]$, which assigns rule $d \in \mathcal{D}$ to user $i \in \mathcal{I}_t$. If $y_{d,i}(t) = 1$, rule d is assigned to user i; otherwise, $y_{d,i}(t) = 0$. Constraints (1.f) ensure that each active user is prioritized by using one scheduling rule each TTI.
- Resource allocation variable: $z_{i,j}[t]$, which indicates whether RB $j \in \mathcal{J}$ is allocated to user $i \in \mathcal{I}_t$. If $z_{i,j}(t) = 1$, RB j is allocated to user i; otherwise, $z_{i,j}(t) = 0$. Constraints (1.g) ensure that at most one RB is allocated to each active user in every TTI.

When applied in accordance with the dynamic conditions of the network and traffic, these variables collectively frame the optimization problem, enabling decisions that effectively balance fairness, QoS satisfaction, and resource utilization.

IV. Proposed FAIR-Q RL Framework

In general, reinforcement learning (RL) seeks to learn an optimal policy of actions that maximizes a long-term reward. Specifically, the proposed RL framework depicted in Fig. 1 learns from its interactions with the scheduling and other RRM entities to identify a policy of scheduling rules that optimizes rewards related to fairness and QoS satisfaction. At each TTI t, a new state $\mathbf{s} \in \mathcal{S}$ is observed, comprising channel conditions, traffic characteristics, and instantaneous QoS indicators and requirements. Given the large and dynamic nature of this state representation, the framework applies processing functions to derive a more stable and representative version. The RL framework observes the processed state $\mathbf{s} \in \mathcal{S}$ and selects an action $\mathbf{a} \in \mathcal{A}$, which corresponds to a scheduling rule $d \in \mathcal{D}$. At TTI $t+1$, a reward function evaluates the effectiveness of the selected action in terms of fairness and QoS satisfaction, while a new state $\mathbf{s}' \in \mathcal{S}$ is observed. Through multiple state-action-reward-state transitions, the RL controller refines a policy of scheduling rules. This refined policy enables the algorithm to automatically determine the appropriate scheduling rule to apply in each state, thereby maximizing fairness and QoS outcomes.

Given the optimization problem outlined in (1), the RL controller depicted in Fig. 1 is tasked with selecting the decision variable $x_{o,d}$, which determines the objective o to be addressed and the corresponding scheduling rule d to be applied. At each TTI, when the RL controller selects a specific objective to address (e.g., NGMN fairness, rate, delay, or PLR), the scheduling rule recommended by the Rule Selector is applied. This rule is then parameterized by the Parameterization Controller to optimize the multi-objective target for the subsequent TTI. The parameterization of scheduling rules allows the scheduling process to adapt to dynamic network and traffic conditions, ensuring compliance with the NGMN fairness requirement. The scheduling function from (1) can then be further decomposed as:

$$\mathcal{F}_o\Big(\mathcal{Q}_{o,i}, \frac{\lambda_{i,j}}{\Lambda_i}\Big) = f_o(\mathcal{Q}_{o,i}) \cdot \frac{(\lambda_{i,j})^{\beta_t - 1}}{(\Lambda_i)^{\alpha_t}}. \tag{2}$$

Here, f_o represents the QoS component of the scheduling function, while the remaining terms constitute the parameterized version of the GPF rule. This parameterization focuses on dynamically adapting $[\alpha_t, \beta_t]$ at each TTI to achieve the NGMN fairness objective. Therefore, at each TTI, the central controller selects the scheduling rule $d \in \mathcal{D}$ and the parameterization scheme $[\alpha_t, \beta_t]$ to improve the multi-objective revenue in the subsequent TTI in terms of NGMN fairness, Guaranteed Bit Rate (GBR), delay and PLR requirements.

A. States

We define first the state components of each active user $i \in \mathcal{I}_t$ at each TTI given by $\mathbf{s}_i = [\mathbf{cqi}_i, \mathbf{q}_i]$, where $\mathbf{cqi}_i$ represents the Channel Quality Indicator (CQI) vector, and $\mathbf{q}_i = [q_{o,i}, \bar{q}_{o,i} - q_{o,i}]$ denotes the Key Performance Indicator (KPI) vector. This KPI vector contains the QoS indicators $q_{o,i}$ and their deviations from the QoS requirements $\bar{q}_{o,i}$ for each objective $o = 1, 2, \ldots, O$. Aggregating CQI reports from all active users within one TTI yields to $\mathbf{cqi} = [\mathbf{cqi}_i]$, $i = 1, 2, \ldots, I_t$. Similarly, regrouping all $\mathbf{q}_i$ vectors by each objective $o \in \mathcal{O}$ gives $\mathbf{q}_o = [q_{o,i}, \bar{q}_{o,i} - q_{o,i}]$, $i = 1, 2, \ldots, I_t$. Combining these vectors across all objectives provides a comprehensive representation of KPI vector $\mathbf{q} = [\mathbf{q}_1, \mathbf{q}_2, \ldots, \mathbf{q}_O]$. In these conditions, the instantaneous and unprocessed state $\mathbf{s}(t) \in \mathcal{S}$ at each TTI becomes:

$$\mathbf{s} = [\alpha_t, \beta_t, \mathbf{cqi}, \mathbf{q}]. \tag{3}$$

This state representation varies with the number of active users, necessitating compression techniques to reduce its dimensionality, as detailed later. However, the state vector in (3) is utilized by the rule selector controller to provide a comprehensive overview of the satisfaction levels across all objectives at each TTI. In contrast, the parameterization controller requires only a subset of this representation corresponding to the fairness objective ($o = 1$). Consequently, the state for the parameterization controller becomes:

$$\mathbf{s}_1 = [\alpha_t, \beta_t, \mathbf{cqi}, \mathbf{q}_1]. \tag{4}$$

The state representations from (3) and (4) permits to efficiently process information, enabling the application of appropriate parameterization and the selection of the most effective scheduling strategies.

B. Actions

The central controller is tasked with providing the action $\mathbf{a}(t) \in \mathcal{A}$ at each TTI t, which directs the scheduling entity on both the choice of scheduling rule and the parameterization variables. This action is expressed as $\mathbf{a} = [a_r, \alpha_t, \beta_t]$, where $a_r \in \{1, 2, \ldots, D\}$ represents the rule selector's decision. The fairness parameters are updated only when the GPF scheduling rule is selected (f_1), utilizing the parameterization controller's action $\mathbf{a}_p = [\Delta\alpha_t, \Delta\beta_t]$, which specifies the parameter adjustments for the current TTI. When invoked, these actions update the fairness parameters as follows:

$$\begin{cases} \alpha_t = \alpha_{t_o} + \Delta\alpha_t, \\ \beta_t = \beta_{t_o} + \Delta\beta_t, \end{cases} \tag{5}$$

where t_o denotes the timestamp of the last parameters' adjustment. Hence, the rule selector is responsible for achieving all objectives globally, while the parameterization controller specifically ensures adherence to the NGMN constraints.

C. Reward Functions

As observed, the rule selector and the parameterization controllers operate on distinct state representations. Similarly, the reward scheme is tailored to their specific roles: the rule selector aims to maximize a multi-objective function that integrates NGMN fairness, GBR, delay, and PLR metrics. In contrast, the parameterization controller focuses on determining optimal parameters to ensure compliance with the NGMN fairness requirement.

For the NGMN fairness objective, the controller's goal is to align the Cumulative Distribution Function (CDF) with the requirement, ensuring it falls within the so-called feasibility region, represented as $\mathbf{s}_1 \in \mathcal{FS}$ [10]. To the left of this requirement lies the unfair region, $\mathbf{s}_1 \in \mathcal{UF}$, where the solutions prioritize system throughput at the expense of fairness. Conversely, to the right of the feasibility region is the over-fair region, $\mathbf{s}_1 \in \mathcal{OF}$, where overly fair solutions result in significant system throughput degradation. The reward function for fairness ($o = 1$) is therefore defined as [10]:

$$r'_1(\mathbf{s}_1, \mathbf{a}_p) = \begin{cases} r_{uf}, & \mathbf{s}'_1 \in \mathcal{UF}, \\ 1, & \mathbf{s}'_1 \in \mathcal{FS}, \\ r_{of}, & \mathbf{s}'_1 \in \mathcal{OF}, \end{cases} \tag{6}$$

where, $\mathbf{s}'_1$ is the parameterization controller state at TTI $t+1$ and sub-rewards r_{uf} and r_{of} are calculated differently as proposed in [10].

In the case of QoS objectives, we define $r_{o,i}$ as the QoS revenue obtained after applying scheduling rule $d \in \mathcal{D}$ in state $\mathbf{s} \in \mathcal{S}$, expressed as follows:

$$r_{o,i} = \begin{cases} 1 - \frac{\bar{q}_{o,i} - q_{o,i}}{\bar{q}_{o,i}}, & \bar{q}_{o,i} > q_{o,i}, o = 2, \\ 1 - \frac{q_{o,i} - \bar{q}_{o,i}}{q_{o,i}}, & \bar{q}_{o,i} < q_{o,i}, o \in \{3,4\}, \\ 1, & otherwise, \end{cases} \tag{7}$$

where, $o = 2$ corresponds to the GBR objective, which aims to ensure that the average throughput of all users meets or exceeds their requirements. The objectives $o \in \{3,4\}$ represent delay and PLR, respectively, and aim to keep the indicators below their corresponding requirements for all active users at each TTI. The QoS reward across all active users is then calculated based on:

$$r'_o(\mathbf{s}, d) = \begin{cases} \sum_{i=1}^{I_t} r'_{o,i} - \sum_{i=1}^{I_t} r_{o,i}, & \sum_i r'_{o,i} \neq \sum_i r_{o,i}, \\ 1, & otherwise, \end{cases} \tag{8}$$

where reward r'_o calculated at TTI $t+1$ for objective $o \in \{2,3,4\}$ detects the improvement of QoS provision compared to the previous TTI.

When training the parameterization controller to optimally adjust the parameters $[\alpha, \beta]$ in alignment with the NGMN fairness requirement at each TTI, the reward values from (6) are reinforced to guide its learning process. In contrast, the Rule Selector Controller is trained using a distinct reward function that encapsulates the multi-objective nature of the scheduling task. This reward function is defined as:

$$r'(\mathbf{s}, d) = \Big(r'_1 + \sum_{o=2}^{O} r'_o\Big) \Big/ 4. \tag{9}$$

This formulation balances the multi-objective components, integrating fairness and other QoS objectives, to train the scheduling rule selector effectively.

D. RL Functions

We employ reinforcement learning (RL) to derive a set of policies that can effectively select appropriate actions for each scheduler state. A policy, in general, represents the probability of taking a specific action given a particular state. In this approach, we consider two distinct policies, corresponding to two types of actions required in our framework:

$$\pi_r(\mathbf{s}, d) = \mathbb{E}[a_r(t) = d \mid \mathbf{s}(t) = \mathbf{s}], \tag{10.a}$$

$$\pi_p(\mathbf{s}_1, \mathbf{a}_p) = \mathbb{E}[\mathbf{a}_p(t) = \mathbf{a}_p \mid \mathbf{s}_1(t) = \mathbf{s}_1], \tag{10.b}$$

where $\pi_r(\mathbf{s}, d)$ is the probability of selecting scheduling rule $d \in \mathcal{D}$ in state $\mathbf{s} \in \mathcal{S}$, and $\pi_p(\mathbf{s}_1, \mathbf{a}_p)$ is the probability of choosing action $\mathbf{a}_p$ when the parameterization controller is in state $\mathbf{s}_1$. To ensure appropriate decision-making, each controller must train functions that approximate these policies as closely as possible to their optimal forms.

The value function $V : \mathcal{S} \to \mathbb{R}$ in general measures the value of a policy π starting from any initial state. Therefore, we denote two types of value functions written as follows:

$$V_r(\mathbf{s}) = \mathbb{E}_{\boldsymbol{\pi}_r}\left[\sum_{t=0}^{\infty} \gamma^t \cdot r' \mid \mathbf{s}(0) = \mathbf{s}\right], \tag{11.a}$$

$$V_p(\mathbf{s}_1) = \mathbb{E}_{\boldsymbol{\pi}_p}\left[\sum_{t=0}^{\infty} \gamma^t \cdot r'_1 \mid \mathbf{s}_1(0) = \mathbf{s}_1\right], \tag{11.b}$$

where $\gamma \in [0,1]$ is the discount factor and $\gamma^t \cdot r'$ is the discounted reward value from state to state.

The action-value function $Q^{\boldsymbol{\pi}} : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ quantifies the expected value of a policy $\boldsymbol{\pi}$ starting from any initial state $\mathbf{s}(0) \in \mathcal{S}$ and taking any initial random action. All subsequent actions are then determined according to the learned policy, capturing the long-term reward achievable under $\boldsymbol{\pi}$. Thus, the representations of this function for each controller become:

$$Q_r(\mathbf{s}, d) = \mathbb{E}_{\boldsymbol{\pi}_r}\left[\sum_{t=0}^{\infty} \gamma^t \cdot r' \mid \mathbf{s}(0) = \mathbf{s}, a_r(0) = d\right], \tag{12.a}$$

$$Q_p(\mathbf{s}_1, \mathbf{a}_p) = \mathbb{E}_{\boldsymbol{\pi}_p}\left[\sum_{t=0}^{\infty} \gamma^t \cdot r'_1 \mid \mathbf{s}_1(0) = \mathbf{s}_1, \mathbf{a}_p(0) = \mathbf{a}_p\right]. \tag{12.b}$$

Considering the principle of temporal difference learning and building upon the findings in [14], the value function and action-value function can be expressed iteratively between consecutive states as follows:

$$V_r(\mathbf{s}) = r(\mathbf{s}, d) + \gamma \cdot V_r(\mathbf{s}') \tag{13.a}$$

$$Q_r(\mathbf{s}, d) = r(\mathbf{s}, d) + \gamma \cdot Q_r(\mathbf{s}', d') \tag{13.b}$$

$$V_p(\mathbf{s}_1) = r_1(\mathbf{s}_1, \mathbf{a}_p) + \gamma \cdot V_p(\mathbf{s}'_1) \tag{13.c}$$

$$Q_p(\mathbf{s}_1, \mathbf{a}_p) = r_1(\mathbf{s}_1, \mathbf{a}_p) + \gamma \cdot Q_p(\mathbf{s}'_1, \mathbf{a}'_p), \tag{13.d}$$

where d' and $\mathbf{a}'_p$ are the actions of rule selector and parameterization controller, chosen on next states $\mathbf{s}'$ and $\mathbf{s}'_1$, respectively.

The objective is to determine the optimal value functions (V_r^*, V_p^*) and action-value functions (Q_r^*, Q_p^*), where optimality corresponds to achieving the highest expected return when the scheduling process begins at a given initial state and a specific action. In our scenario, the state space for both controllers is infinite and multi-dimensional, while the action space for the parameterization controller is continuous and infinite. Consequently, classical and tabular RL methods are unsuitable as they rely on discrete state and action spaces, making optimality unattainable. Instead of ensuring absolute optimality, our approach focuses on deriving effective approximations of these value functions that are sufficiently close to their optimal counterparts, enabling practical and efficient decision-making under these complex conditions.

E. Approximation of RL Functions

The dependency of the controller states on variability caused by the number of users and system bandwidth is mitigated using the state compression functions proposed in [14]. Despite these compression techniques, the state remains multi-dimensional, making it infeasible to store state-action transitions in regular tabular formats. To address this, we approximate the value and action-value functions using neural networks, enabling efficient representation and learning in high-dimensional state spaces. Let $\tilde{V}_r^*$ and $\tilde{Q}_r^*$ denote the approximations of the value and action-value functions, respectively, for the rule selector controller. These approximations are modeled by the following neural networks:

$$\tilde{V_r^*}(\mathbf{s}) = h_r^v[\theta_t^v, \psi(\mathbf{s})], \tag{14.a}$$

$$\tilde{Q_r^*}(\mathbf{s}, d) = h_r^d[\theta_t^d, \psi(\mathbf{s})], \tag{14.b}$$

where $h_r^v, h_r^1, h_r^2, \ldots, h_r^D$ denote the neural networks used to approximate the value and action-value functions, respectively. The feature vector is represented as $\psi(\mathbf{s})$, and $\theta^v, \theta^1, \theta^2, \ldots, \theta^D$ are the sets of weights that must be optimized during RL training. Consequently, the rule selector has a dedicated neural network corresponding to each scheduling rule. In case of the parameterization controller, we denote by $\tilde{V}_p^*$ and $\tilde{Q}_p^*$ the approximated value and action-value functions, respectively, through the following neural networks:

$$\tilde{V_p^*}(\mathbf{s}_1) = h_p^v[\theta_t^v, \psi(\mathbf{s}_1)], \tag{15.a}$$

$$\tilde{Q_p^*}(\mathbf{s}_1, d) = h_p^q[\theta_t^q, \psi(\mathbf{s}_1)], \tag{15.b}$$

where h_p^v and h_p^q are the neural networks approximating the value and action-value functions, respectively, for the parameterization controller. The action-value function is realized using a single neural network, which outputs a two-dimensional continuous action $[\Delta\alpha_t, \Delta\beta_t]$ at each TTI, effectively tuning the scheduling parameters dynamically.

F. Training the Controllers

The structure of the neural network is characterized by the number of layers and the number of hidden nodes in each layer. Let L represent the total number of layers in the network, and N_l denote the number of nodes in layer $l \in \{1, 2, ..., L\}$. The number of nodes in the input and output layers is predefined: N_1 corresponds to the dimensionality of the state space, while $N_L = 1$ for the rule selector neural network and $N_L = 2$ for the parameterization controller. However, the architecture of the hidden layers (specifically, the number of hidden layers $L-2$ and the number of nodes in each hidden layer) must be determined through cross-validation procedures to optimize performance.

The sets of weights $\theta_r^v, \theta_r^1, \theta_r^2, \ldots, \theta_r^D$ or θ_p^v, θ_p^q are interconnecting the nodes between successive layers in the neural network. Let $\mathbf{W}_l = \{w_{b,m}, b = 1, \ldots, N_l, m = 1, \ldots, N_{l+1}\}$ denote the matrix of weights connecting layer l to layer $l+1$. The number of weights to be tuned during the learning phase between these layers is $(N_l + 1) \times N_{l+1}$. The compressed controllable states $\mathbf{s} \in \mathcal{S}$ propagate through the network, undergoing non-linear transformations at each layer. The errors, calculated as the difference between the network's learned values and their ground truth (or the reinforced values derived from (13.a) - (13.d)), are backpropagated through the network. This process, applied to each neural network type as described in [10] and [14], updates the weights at each layer and node based on the gradient descent principle.

G. RL Algorithms

The RL algorithms used in this paper vary for each controller. However, in general, when training RL with neural network approximations, the choice of algorithm depends on how target values are computed and how output layer errors are estimated [10], [14]. In the case of rule selector, the target values for a given state $\mathbf{s}' \in \mathcal{S}$ are computed using (13.a) and (13.b). Specifically, $\mathbf{s}'$ is propagated through the network to obtain an approximated value or action-value, which is then updated with the observed reward. Subsequently, the previous state $\mathbf{s} \in \mathcal{S}$ is propagated through the network to compute another approximation of the value or action-value. The manner in which the error is calculated based on these approximations for the current $(\mathbf{s}')$ and previous $(\mathbf{s})$ states determines the choice of RL algorithm employed [14].

The parameterization controller is updated less frequently than the rule selector, with its errors reinforced only when the simple GPF rule ($d = 1$) is selected. However, the target values for network approximations are computed similarly, using (13.c) and (13.d). The errors are determined at each TTI based on the learning experiences $\{\mathbf{s}_1, \mathbf{a}_p, r_1, \mathbf{s}_1'\}$ [10].

V. Simulation Results

We consider downlink transmission with a system bandwidth of 20MHz and $J = 100$ RBs to be shared among active users with varying network and traffic conditions. The number of active users $I_t \in \{15, 120\}$ is randomly adjusted each one second to have high diversity of status change from active to idle states and vice-versa. The user speed is 120kmph with fast-fading Jakes model to experience high diversity in CQI reports in both time and frequency domains. The CQI report is errorless, periodic and full-band in order to have complete information about the CQI statistics for each active user $i \in \mathcal{I}_t$. The rest of physical layer parameters are imported from the 3GPP simulation scenarios [14]. The RLC layer is modeled through ARQ ACK mode with 5 maximum retransmissions in case of lost packets. Experimental results were conducted by using the LTE-Sim [17] equipped with data processing and ML algorithms as developed in [18].

TABLE I: Mean Percentages of Satisfied Users with Fairness and QoS Objectives

Scheduling Rule	**50%**	**55%**	**60%**	**65%**	**70%**	**75%**	**80%**	**85%**	**90%**	**95%**	**100%**
GPF & CACLA2	9.288	9.037	8.67	8.012	6.502	3.332	2.183	0.759	0.235	0.048	0.005
BF	23.764	18.793	15.891	8.392	3.049	0.842	0.554	0.066	0.004	0	0
RAD	22.524	18.736	16.32	9.424	4.004	1.349	0.917	0.127	0.011	0	0
mM	5.35	4.794	4.544	4.002	1.266	1.185	1.18	1.124	1.038	0.654	0.654
LM	25.283	21.818	19.995	14.801	9.001	4.51	3.226	1.053	0.32	0.062	0.032
EDF	17.476	15.718	14.548	10.93	6.566	3.327	2.462	0.687	0.094	0.008	0.007
MLWDF	12.391	12.116	11.966	11.742	11.623	11.445	11.328	11.065	10.746	9.946	9.353
LOG	23.041	22.368	21.93	21.491	21.101	20.7	20.343	19.914	19.03	15.979	13.135
EXP1	20.329	19.611	18.838	18.109	17.324	15.84	14.734	11.546	6.942	2.671	1.713
EXP2	24.752	20.87	18.468	11.167	4.753	1.471	0.931	0.095	0	0	0
MDU	24.238	23.577	23.138	22.395	21.54	20.621	20.308	19.47	18.164	15.402	13.289
PLF	0.291	0.118	0.093	0.039	0.011	0.003	0.002	0	0	0	0
OPLF	0.87	0.505	0.406	0.238	0.126	0.055	0.049	0.018	0.004	0.001	0
RL Algorithm	**50%**	**55%**	**60%**	**65%**	**70%**	**75%**	**80%**	**85%**	**90%**	**95%**	**100%**
Q	26.706	26.564	26.455	26.306	26.118	25.777	25.482	24.258	20.817	15.368	12.885
DQ	21.614	21.221	20.952	20.566	20.092	19.411	19.107	17.756	15.156	10.803	9.266
SARSA	22.523	22.375	22.268	22.147	22.024	21.919	21.818	21.664	21.356	20.561	19.599
QV	18.815	17.898	17.331	15.785	13.103	9.32	7.858	4.172	1.508	0.399	0.243
QV2	28.767	28.295	27.924	27.568	27.217	26.671	26.143	24.614	20.5	13.982	10.705
QVMAX	23.851	21.961	20.71	18.828	17.133	15.208	14.508	11.926	8.947	5.717	4.812
QVMAX2	19.526	18.53	18.18	16.668	14.054	10.453	9.982	5.477	2.026	0.535	0.209
ACLA	20.127	19.688	19.342	18.911	18.525	17.975	17.777	16.891	15.364	12.153	11.235

Each active user requests a Variable Bit Rate (VBR) service type with a packet size given by a Pareto distribution and the arrival rate in MAC queue with a geometric distribution [14]. To increase the dynamics of network conditions, we vary the QoS requirements each 1000 TTIs in the following ranges of values: GBR requirement ($o = 2$) $\bar{q}_2 \in \{32, 64, 128, 256, 512, 1024\} kbps$; delay requirement ($o = 3$) $\bar{q}_3 \in \{50, 100, 150, 200, 250, 300\} ms$; PLR requirement ($o = 4$) $\bar{q}_4 \in \{10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}\}$.

A wide pallette of scheduling rules is used to train the rule selector controller and achieve the multi-objective target. For fairness objective ($o = 1$), we consider the GPF rule with the double parameterization $[\alpha_t, \beta_t]$ decided by the parameterization controller. For the GBR objective ($o = 2$), we consider four scheduling rules [18]: Barrier Function (BF), Required Activity Detection (RAD), minimum/Maximum rate (mM), Langrange Multiplier (LM). The scheduling rules focused on packet delay minimization ($o = 3$) are [18]: Earliest Due to Date Function (EDF), Modified Largest Weighted Delay First (MLWDF), Logarithmic Function (LOG), Exponential Function 1 (EXP1), Exponential Function 2 (EXP2) and Max-Delay Utility (MDU). The PLR objective ($o = 4$) is represented by Packet Loss Fair (PLF) and Opportunistic Packet Loss Fair (OPLF) [18]. All these algorithms are applied in parallel by using the same network and traffic conditions relying on comprehensive comparison of the obtained results.

When training the rule selector controller to apply the most appropriate scheduling rule on each state to meet the feasibility NGMN region while respecting the dynamic QoS requirements, different RL algorithms are implemented [18]: Q-Learning, Double-Q-Learning, SARSA, QV, QV2, QVMAX, QVMAX2, and ACLA. Each algorithm is trained for a period of 500 seconds by using the same network and traffic conditions. The value and action-value neural networks consider a number $L = 3$ layers with $N_2 = 150$ hidden nodes and tangent hyperbolic activation functions. The parameterization controller is trained based on CACLA2 algorithm [10] in the same time with the rule selector, meaning that the training samples are provided only when the fairness objective is addressed and the GPF rule is selected. Even in these conditions, CACLA2 shows very well convergence properties for a structure of neural networks with $L = 3$ with $N_2 = 100$ hidden nodes. Once trained, both controllers are evaluated across ten independent simulations, each with a duration of 100 seconds, and the results are averaged.

To evaluate the effectiveness of the trained RL algorithms, we use a performance metric defined as the percentage of TTIs during which the NGMN fairness and QoS constraints are met for a specified proportion of users (x%). Specifically, we denote this metric as p_x, which represents the mean percentage of TTIs where the multi-objective target is achieved for at least x% of users. This approach enables us to comprehensively assess the performance of each scheduling rule and the trained RL policies by observing how effectively they balance fairness and QoS requirements across various user satisfaction thresholds (x%). By using this metric, we gain insights into the adaptability and robustness of the scheduling strategies in meeting diverse network demands.

Table I presents the performance metrics for all evaluated scheduling rules and RL-based policies. The GPF scheduling rule, parameterized using the CACLA2 RL algorithm, shows relatively low percentages of TTIs satisfying all four objectives, as it primarily focuses on meeting the feasibility requirements of the NGMN fairness criterion. BF and RAD scheduling rules deliver comparable performance across metrics but experience an 11% drop in satisfaction when moving from x=60% to x=70% satisfied users. The mM scheduling rule performs poorly in achieving the multi-objective target, consistent with previous findings that it operates effectively only under low-traffic conditions [18]. Among

GBR-oriented scheduling rules, the LM metric exhibits a notable 4% improvement at p_{75} compared to other GBR-focused metrics. For delay-oriented scheduling rules, LOG and MDU perform the best, with p_{100} values exceeding 13%, indicating that these rules are the most effective in satisfying all objectives under high user satisfaction thresholds. EXP1 and EXP2 demonstrate similar performance up to p_{65}, after which their satisfaction rates decline significantly. Packet-loss-oriented scheduling rules, which prioritize users with high retransmission counts and poor network conditions, negatively impact fairness and other QoS metrics. When combining these scheduling rules within RL-driven policies, the outcomes differ significantly. Among the RL algorithms, QV2 provides the highest proportion of TTIs satisfying the multi-objective target for 50% to 85% of users. However, SARSA achieves the best results for higher satisfaction thresholds (90%, 95%, 100%), with $p_{100} = 19.6\%$, surpassing the best standalone scheduling rule (MDU) by over 6%. This highlights SARSA's ability to maintain consistent performance under stringent QoS and fairness requirements, ensuring that nearly 20% of the scheduling sessions satisfy all objectives for all users.

VI. Conclusions

In this paper, we propose FAIR-Q, a multi-objective RL framework designed to simultaneously meet four critical objectives: NGMN fairness, throughput, delay, and packet loss requirements. FAIR-Q incorporates two controllers: a Rule Selector Controller, which dynamically selects the most appropriate scheduling rule based on real-time network and traffic conditions, and a Parameterization Controller, which optimally adjusts the parameters of the GPF rule to address fairness deficits under varying traffic loads. Each controller is trained using dedicated RL algorithms and leverages neural networks that can be extended to deeper architectures, enabling the exploration of additional scenarios to effectively balance multi-objective performance. Our implementation demonstrates that SARSA learning produces the most effective policies for scheduling rule selection, while CACLA2 excels in parameterizing the GPF rule to ensure adherence to NGMN fairness requirements. Simulation results reveal that more than 15% of the scheduling time achieves a state where all users meet the proposed multi-objective criteria.

The FAIR-Q framework seamlessly integrates into the evolving 6G landscape by addressing its stringent QoS and fairness demands. As 6G networks promise dynamic and heterogeneous traffic scenarios—ranging from ultra-low-latency applications to massive IoT deployments—the adaptability of FAIR-Q ensures alignment with the needs of future network operations. By incorporating advanced scheduling strategies and adaptive parameterization mechanisms, FAIR-Q enables scalable and intelligent RRM, making it well-suited for next-generation multimedia services, mission-critical applications, and diverse user demands. The modular RL-based design of FAIR-Q facilitates integration into Open RAN architectures, which are expected to play a pivotal role in 6G networks. This adaptability positions FAIR-Q as a robust and forward-thinking solution to the complex challenges of multi-objective resource allocation in 6G environments.

Acknowledgment

This research was conducted as part of the UKIERI-SPARC project "DigIT—Digital Twins for Integrated Transportation Platform", grant number UKIERI-SPARC/01/23.

References

[1] C.-X. Wang, X. You, X. Gao, X. Zhu, Z. Li, C. Zhang, H. Wang, Y. Huang, Y. Chen, H. Haas, J. S. Thompson, E. G. Larsson, M. D. Renzo, W. Tong, P. Zhu, X. Shen, H. V. Poor, and L. Hanzo, "On the Road to 6G: Visions, Requirements, Key Technologies, and Testbeds," *IEEE Comms. Surveys Tutorials*, vol. 25, no. 2, pp. 905–974, 2023.

[2] M. Polese, L. Bonati, S. D'Oro, S. Basagni, and T. Melodia, "Understanding O-RAN: Architecture, Interfaces, Algorithms, Security, and Research Challenges," *IEEE Communications Surveys Tutorials*, vol. 25, no. 2, pp. 1376–1411, 2023.

[3] M. Zangooei, N. Saha, M. Golkarifard, and R. Boutaba, "Reinforcement Learning for Radio Resource Management in RAN Slicing: A Survey," *IEEE Communications Magazine*, vol. 61, no. 2, 2023.

[4] A. Abouaomar, A. Taik, A. Filali, and S. Cherkaoui, "Federated Deep Reinforcement Learning for Open RAN Slicing in 6G Networks," *IEEE Communications Magazine*, vol. 61, no. 2, pp. 126–132, 2023.

[5] I. S. Comșa, M. Aydin, S. Zhang, P. Kuonen, and J.-F. Wagen, "Reinforcement Learning Based Radio Resource Scheduling in LTE-Advanced," in *17th International Conference on Automation and Computing*, 2011.

[6] I. S. Comsa, M. Aydin, S. Zhang, P. Kuonen, and J. Wagen, "Multi Objective Resource Scheduling in LTE Networks Using Reinforcement Learning," *International Journal of Distributed Systems and Technologies (IJDST)*, vol. 3, no. 2, pp. 1 – 19, 2012.

[7] F. Al-Tam, N. Correia, and J. Rodriguez, "Learn to Schedule (LEASCH): A Deep Reinforcement Learning Approach for Radio Resource Scheduling in the 5G MAC Layer," *IEEE Access*, vol. 8, pp. 108 088 – 108 101, 2020.

[8] I. S. Comșa, M. Aydin, S. Zhang, P. Kuonen, J.-F. Wagen, and Y. Lu, "Scheduling Policies Based on Dynamic Throughput and Fairness Tradeoff Control in LTE-A Networks," in *39th Annual IEEE Conference on Local Computer Networks*, 2014, pp. 418–421.

[9] I.-S. Comșa, S. Zhang, M. Aydin, J. Chen, P. Kuonen, and J.-F. Wagen, "Adaptive Proportional Fair Parameterization Based LTE Scheduling Using Continuous Actor-Critic Reinforcement Learning," in *IEEE Global Communications Conference (GLOBECOM)*, 2014, pp. 4387–4393.

[10] I.-S. Comșa, S. Zhang, M. Aydin, P. Kuonen, R. Trestian, and G. Ghinea, "A Comparison of Reinforcement Learning Algorithms in Fairness-Oriented OFDMA Schedulers," *Information*, vol. 10, no. 10, 2019.

[11] X. Du, T. Wang, Q. Feng, C. Ye, T. Tao, L. Wang, Y. Shi, and M. Chen, "Multi-Agent Reinforcement Learning for Dynamic Resource Management in 6G in-X Subnetworks," *IEEE Transactions on Wireless Communications*, vol. 22, no. 3, pp. 1900–1914, 2023.

[12] S. Mhatre, F. Adelantado, K. Ramantas, and C. Verikoukis, "Intelligent QoS-aware Slice Resource Allocation with User Association Parameterization for Beyond 5G O-RAN-based Architecture Using DRL," *IEEE Transactions on Vehicular Technology*, pp. 1–14, 2024.

[13] H. Zhou, M. Erol-Kantarci, and H. Vincent Poor, "Learning From Peers: Deep Transfer Reinforcement Learning for Joint Radio and Cache Resource Allocation in 5G RAN Slicing," *IEEE Trans. on Cognitive Comms. and Networking*, vol. 8, no. 4, pp. 1925–1941, 2022.

[14] I.-S. Comșa, S. Zhang, M. E. Aydin, P. Kuonen, Y. Lu, R. Trestian, and G. Ghinea, "Towards 5G: A Reinforcement Learning-Based Scheduling Solution for Data Traffic Management," *IEEE Transactions on Network and Service Management*, vol. 15, no. 4, pp. 1661–1675, 2018.

[15] I.-S. Comșa, A. De-Domenico, and D. Ktenas, "QoS-Driven Scheduling in 5G Radio Access Networks - A Reinforcement Learning Approach," in *IEEE Global Communications Conference (GLOBECOM)*, 2017, pp. 1–7.

[16] I.-S. Comșa, A. Molnar, I. Tal, P. Bergamin, G.-M. Muntean, C. H. Muntean, and R. Trestian, "A Machine Learning Resource Allocation Solution to Improve Video Quality in Remote Education," *IEEE Transactions on Broadcasting*, vol. 67, no. 3, pp. 664–684, 2021.

[17] G. Piro, L. A. Grieco, G. Boggia, F. Capozzi, and P. Camarda, "Simulating LTE Cellular Systems: An Open-Source Framework," *IEEE Trans. on Vehicular Nets.*, vol. 60, no. 2, pp. 498 – 513, 2011.

[18] I.-S. Comșa, *Sustainable Scheduling Policies for Radio Access Networks Based on LTE Technology*. Ph.D. dissertation, School Comput. Sci. Technol., Univ. of Bedfordshire, Luton, U.K., 2014.