

# Towards User Scheduling for 6G: A Fairness-Oriented Scheduler Using Multi-Agent Reinforcement Learning

Mingqi Yuan<sup>1</sup>, Qi Cao, Man-On Pun, *Senior Member IEEE* and Yi Chen

**Abstract**—User scheduling is a classical problem and key technology in wireless communication, which will still play an important role in the prospective 6G. There are many sophisticated schedulers that are widely deployed in the base stations, such as *Proportional Fairness* (PF) and *Round-Robin Fashion* (RRF). It is known that the *Opportunistic* (OP) scheduling is the optimal scheduler for maximizing the average user data rate (AUDR) considering the *full buffer* traffic. But the optimal strategy achieving the highest fairness still remains largely unknown both in the full buffer traffic and the *bursty* traffic. In this work, we investigate the problem of fairness-oriented user scheduling, especially for the RBG allocation. We build a user scheduler using *Multi-Agent Reinforcement Learning* (MARL), which conducts distributional optimization to maximize the fairness of the communication system. The agents take the cross-layer information (e.g. RSRP, Buffer size) as *state* and the RBG allocation result as *action*, then explore the optimal solution following a well-defined *reward* function designed for maximizing fairness. Furthermore, we take the *5%-tile user data rate* (STUDR) as the key performance indicator (KPI) of fairness, and compare the performance of MARL scheduling with PF scheduling and RRF scheduling by conducting extensive simulations. And the simulation results show that the proposed MARL scheduling outperforms the traditional schedulers.

**Index Terms**—User scheduling, RBG allocation, Fairness-oriented, Multi-Agent reinforcement learning

## I. INTRODUCTION

FOR the 6th generation wireless communication technology, both of the basic architectures and performance components remain largely undefined [1]. But it is certain that the 6G will undergo an unprecedented revolution, which will make it immensely different from the former wireless cellular systems [2]. The utilization of 5G greatly improves the communication services in densely-populated cities, especially for the indoor communication scenarios. However, there are still a large number of population can not obtain the basic data services, such as the developing countries and rural areas [3]. So the 6G is expected to construct a wireless network which

is both vertical and horizontal to benefits more population rather than the majority people in cities. Providing such service depends largely on the new network resource scheduling technology, especially the fairness-oriented scheduling algorithms.

In this work, we focus on the user scheduling, especially the Resource Block Groups (RBGs) allocation. But we consider the fairness-oriented scheduling and aim to build a scheduler which maximizes the fairness of the communication system. There are many traditional model-based schedulers which consider the system fairness, such as Proportional Fairness (PF) and Round-Robin Fashion (RRF) [4]. It is known that the Opportunistic (OP) scheduling is the optimal solution, which maximizes the average user data rate (AUDR) considering the full buffer traffic [5]. The OP scheduling follows the greedy policy, and always allocates advantageous resources to the users with the highest expected throughput. However, the optimal strategy achieving the highest fairness is still unknown both in the full buffer traffic and the bursty traffic. It is generally considered impossible to solve this problem using mathematical modeling. And the technical challenges can be summarized as the following two aspects.

On the one hand, the definition and evaluation metrics of fairness are still debatable until today. Intuitively, whether all the requests of users can get responses equally is a significant definition of fairness. To that end, the RRF is a perfect solution which allocates all RBGs to each user in turns regardless of their status [6]. But this operation seems like to be unfair to all the users considering the long-time allocation. Because it greatly decreases the global transmission rate of the communication system, and that is intolerable for the resource cost. Thus the pursuit of fairness should be based on the full use of system resources, which is followed by the PF scheduling. [7]. Except for the definition of fairness, the evaluation metrics also have diverse forms. A commonly accepted metric is the 5%-tile user data rate (STUDR), which reflects the fairness from the perspective of transmission rate. Furthermore, we can calculate the Jain's Fairness Index (JFI) with individual user data rates [8]. Considering the sequence  $x = x_1, x_2, \dots, x_n$ , then the JFI is calculated as  $\mathcal{J}(x_1, x_2, \dots, x_n) = \bar{x}^2 / x^2$ . But this method overstates the importance of ratio, and ignores the actual transmission rate, because the JFI will get the maximum value as long as the  $x_1 = x_2 = \dots = x_n$ .

On the other hand, a wireless communication network is highly complex with many components and mechanisms, rendering the task of mathematically modeling the whole system analytically intractable. Most of the traditional scheduling

This work was supported, in part, by the Shenzhen Science and Technology Innovation Committee under Grant No. ZDSYS20170725140921348 and JCYJ20190813170803617, and by the National Natural Science Foundation of China under Grant No. 61731018. (*Corresponding author: Man-On Pun.*)

M. Yuan, Q. Cao, M. Pun and Y. Chen are with the School of Science and Engineering, The Chinese University of Hong Kong, Shenzhen, 518172, China (e-mail: mingqi.yuan@link.cuhk.edu.cn; caoqi@cuhk.edu.cn; simonpun@cuhk.edu.cn; yichen@cuhk.edu.cn.)

M. Yuan, Q. Cao and M. Pun are with the Shenzhen Key Laboratory of IoT Intelligent Systems and Wireless Network Technology, 518172, China

M. Yuan, M. Pun and Y. Chen are with the Shenzhen Research Institute of Big Data, 518172, China

algorithms are designed based on single layer of the OSI protocol, which can not consider the cross-layer information and apply global optimization. Thus it is difficult to achieve higher fairness through the model-based approaches. Recently years, the machine learning (ML) methods are introduced to the research of wireless communication [9]. In sharp comparison with the model-based approach, the ML-based method is data-driven which applies optimization by exploiting the massive data in the network. This allows the ML-based method to solve many network optimization problems without establishing the mathematical model. Moreover, the utilization of the Deep Neural Network (DNN) makes it easy to use diverse network data to build models. For instance, Cao et al. designed a DNN-based framework to predict network-level system performance which take the cross-layer network data as input [10]. And the simulation results show that the model can make accurate predictions at the cost of high computational complexity. For user scheduling, another method entitled Reinforcement Learning (RL) is also introduced to improve the existed schedulers or build new schedulers. RL is one of the most representative leanings methods of machine learning, which aims to maximize the long-term expected return by learning from the interactions with environment [11]. In RL, the agent accepts the state of environment and make corresponding action, then the agent will be rewarded or punished. To get higher reward, the agent must continuously adjust its policy to make better action. The RL has strong exploration ability, and always achieve surprising performance. There are much pioneering work have already been achieved with RL.

For instance, Comsa et al. suggests that the PF scheduling cannot adapt to dynamic network conditions because of the settled scheduling preference [12]. Thus they proposes a parameterized PF scheduling entitled Generalized PF (GPF) by using the deep RL approach. The GPF collects diverse network data such as Channel Condition Indicator (CQI) and user throughput, then adaptively adjust the scheduling preference in each transmission time interval (TTI). As a result, the GPF is showed to be more flexible in handling dynamic network conditions. Furthermore, Zhang et al. builds an RL-enabled algorithm selector, which scans the network status and intelligently select the best scheduling algorithm from a set of scheduling algorithms in each TTI [13]. And the simulation results show that the fusion strategy achieves higher performance compared to the traditional methods. However, both of the two algorithms have low efficiency because of the huge computation, and they are not independent schedulers. Therefore, Xu et al. considers building entirely independent schedulers with RL method [14], and proposes an RL-based scheduler. The scheduler takes the estimated instantaneous data rate, the averaged data rate and other metrics as the input of the agent, then outputs the allocation preference for each user. The performance comparison in [14] indicates that the RL-based scheduler can effectively improve the throughput, along with the fairness and the packet drop rate. But this scheduler can only allocate RBGs in full buffer traffic, because it takes fully-connected neural network as the backbone of agent. Similarly, Hua et al. investigates the resource management in network slicing, and proposes a GAN-powered deep

distributional RL method to find optimal solution of demand-aware resource management [15]. The agent takes the number of arrived packets in each slice within a specific time window as input, then generates bandwidth allocation to each slice. And the simulations results show that method significantly improve the utilization efficiency of resources. In this paper, the problem is also solved with RL method, but the Multi-Agent Reinforcement Learning (MARL) fashion is adopted in our work.

In this work, we consider a Long-Term Evolution (LTE) network with one base station (BS) scheduling its RBGs to multiple users in bursty traffic. Take the allocation result of RBGs as the action in RL, so multi RBGs produce a joint action. For single agent RL, it is difficult to learn a joint policy because of the exponential action space. Thus the MARL is introduced to address this problem, which can applies distributional optimization. In MARL, there are multiple agents interacting with the environment [16]. From the perspective of Game Theory, their relationship can be summarized as Cooperative, Zero-Sum and General-Sum [17]. The Cooperative means all the agents receive the same reward and aim to maximize the same objective. The Zero-Sum means all the agents is fully competitive, and the gain of one agent is the loss of other agents. Thus a middle situation is General-Sum, it means all the agents are neither fully cooperative nor entirely adversarial. As a distributional optimization methods, the optimal solution of MARL can be defined as a Nash Equilibrium. Inspired by the distributional setting and powerful exploration abilities of RL, the MARL offers a solution paradigm that can cope with complex problems. One of the most successful applications is AlphaStar, which is a game AI for the StarCraft II using MARL [18]. The AlphaStar can achieve grandmaster level performance in the confrontation with mankind, which implies the great potential of MARL.

In this work, we let each agent be responsible for a single RBG, then build a MARL system to learn a scheduling policy which maximizes the system fairness. All the agents are set to be fully cooperative and accept the specific cross-layer network data as input. Furthermore, we fully discuss the design of the reward function. In RL system, the reward function characterizes the optimization objective, but there are no theoretical research about how to accurately and properly design a reward function due to the complex and changeable task. For the RBG allocation, the performance is affected by multiple factors, so that it becomes more difficult to get the reward function. Through a large number of experiments and analysis, a feasible reward function is successfully found which can effectively achieve our objective. Finally, we compare the performance of the MARL scheduler with the PF scheduling and RRF scheduling. And the simulations results shows our method outperforms the two traditional model-based algorithms. The main contributions of this paper are as follows:

- We analyze the necessity of fairness-oriented scheduling algorithm in the development of 6G, which is expected to provide widespread network service to larger population. Furthermore, the main challenges of building such algorithms are fully discussed, including the debatable definition of fairness and the limitation of the traditional model-

based approaches. Then we summarize the applications of data-driven approaches in wireless communication, especially the wireless resource management;

- To find the optimal fairness-oriented scheduling policy, we build a LTE downlink wireless network to simulate the realistic communication scenarios, select a set of cross-layer network data as the state of the communication system, then build a MARL system to learn the objective policy. In this system, all the agents are set to be fully cooperative and maximize the fairness together;
- The reward function determines whether the agents can learn the expected policy. To find the feasible reward function, we conduct a large amount of experiments and analysis. Then we successfully develop a fairness-oriented reward function, which is shown to be effective in realizing our objective;
- In contrast to most works in the literature that focus on the full-buffer transmission, we consider the more realistic bursty traffic in this work. To cope with the dynamic network conditions such as the time-varying number of active users, an adaptive working mechanism of the agents is proposed;
- Finally, extensive simulations are conducted to demonstrate the better performance of the MARL scheduler over the traditional schedulers. Furthermore, we analyze the special scheduling policy of the MARL scheduler in depth by data analysis, and provide numerical results about the implementation details.

The remainder of the paper is organized as follows: Section II gives the problem formulation and defines the system model. Section III elaborates some essential mathematical backgrounds of MARL. Then Section IV talks about the MARL for RBG allocation in detail. And Section V shows the simulation results and numerical analysis. Finally, Section VI summarizes the paper and proposes the prospects.

\*\*\*\*\*

## II. SYSTEM MODEL AND PROBLEM FORMULATION

### A. System Model

We consider a single-cell LTE wireless communication network in which the BS serves multiple user equipments (UEs) in the downlink in the Frequency Division Duplexing (FDD) mode. The UE arrival is modeled as a Poisson process with an arrival rate  $\lambda$ . Upon arrival, each UE requests a different but finite amount of traffic data that is stored in the buffer at the transmitter. After its request is completed, the corresponding UE departs the network immediately, which effectively simulates the bursty traffic mode. Furthermore, the frequency resources are divided into RBGs. We consider the most common network setting that one RBG can be allocated to at most one UE in each TTI as shown in Fig. 1. In what follows, some critical mechanisms are elaborated.

1) *Out Loop Link Adaptation*: On the user side, CQI levels represent the received signal-to-noise ratio (SNR) on each resource block (RB), and they are periodically computed and reported. Thus the received instantaneous SNR directly influences the selection of MCS. However, it should be noted

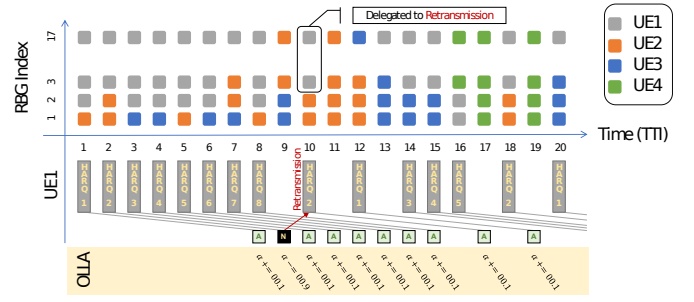


Fig. 1. Network Mechanisms

that there is a CQI offset reflecting the detection sensitivity per user which is unknown to the BS. Thus, to compensate for the discrepancy between the chosen MCS and the optimal MCS for different users. An OLLA process is carried out on BS to offset a CQI value  $q$ , which is

$$\bar{q} = [q + \alpha], \quad (1)$$

where  $[\cdot]$  is the rounding operation and  $\bar{q}$  is the offset CQI readily for MCS selection. Besides,  $\lambda$  is the adjustment coefficient given by

$$\alpha = \begin{cases} \alpha + s_A, & \mathfrak{A} = 1, \\ \alpha + s_N, & \mathfrak{A} = 0, \end{cases} \quad (2)$$

where  $\mathfrak{A}$  is the corresponding ACK/NACK feedback with “1” indicating a successful transmission while “0” a failed transmission, and the update rate  $s_A > 0$  and  $s_N < 0$  can be customized.

2) *Transmission Block Formation*: Let  $\mathcal{F}(q)$  denote the mapping from the CQI  $q$  to its spectral efficiencies (SE), and  $\mathcal{G}$  denote a set of CQI levels measured by a user in terms of a group of RBs. The supportive number of bits that can be loaded to the group of RBs for the user is given by

$$f(\mathcal{G}) = |\mathcal{G}| \cdot \mathcal{F}\left(\left\lfloor \frac{1}{|\mathcal{G}|} \sum_i \{\mathcal{G}\}_i \right\rfloor\right) \quad (3)$$

where  $\lfloor \cdot \rfloor$  is the floor function. For instance, the estimated data rate of user  $n$  in RBG  $k$  can be obtained by  $R_{n,k} = f(\mathcal{G}_n(k))$ , where  $\mathcal{G}_n(k)$  is the set of all RBs in RBG  $k$  measured by user  $n$ . When multiple RBGs are allocated to the user, they are utilized to convey one transport block (TB) with the same MCS. The size of data loaded to the TB is given by  $T_n = f(\mathcal{G}_n)$ , where  $\mathcal{G}_n$  is the set of all RBs allocated to user  $n$ .

3) *Hybrid Automatic Repeat reQuest and Retransmission*: When a TB is packed for a user, the corresponding data will be loaded to a HARQ buffer and cleared out from the data buffer. The BS can arrange at most eight HARQ processes (each with a HARQ buffer) for each user, and will see an ACK/NACK message from the corresponding user in seven TTIs. In the case of ACK, the HARQ process terminates, while in the case of NACK, a retransmission is triggered and the HARQ process is still held. The RBGs initially delegated to the first transmission will conduct the retransmission, thus being unavailable for user scheduling temporarily. The MCS

selection and TB size remain the same for the retransmission and the HARQ process will expire at five times of consecutive failure, which literally causes the so-called packet loss.

### B. Problem Formulation

1) *Key Performance Indicators*: We denote by  $t_{n,a}$  and  $t_{n,d}$  the arrival and departure time for a user indexed by  $n$ , respectively. Then the corresponding user data rate at time  $t \in [t_{n,a}, t_{n,d}]$  can be defined as:

$$\psi_t^n = \frac{\sum_{i=t_{n,a}}^t T_n[i] \mathfrak{A}_n[i]}{\min(t, t_{n,d}) - t_{n,a}}, \quad (4)$$

where  $T_n[i]$  is the packet size transmitted during the  $i$ -th TTI. Denote by  $N_t$  the total number of users that arrive during a time period of TTI=1, ...,  $t$ . The AUDR can be defined as

$$D(t) = \frac{1}{N_t} \sum_{n=1}^{N(t)} \psi_t^n. \quad (5)$$

Let  $\phi_t$  denote the 5%-tile user data rate, which can be written as:

$$\phi_t = \mathfrak{T}(\Psi_t), \Psi_t = (\psi_t^1, \dots, \psi_t^{N_t}) \quad (6)$$

where  $\mathfrak{T}(\cdot)$  is the searching operator for getting the user data rate of the worst 5% users.

2) *Optimization objective*: Consider a network defined above with a set  $K$  of RBGs, the Algo. 1 elaborates the optimization objective.

---

#### Algorithm 1 Optimization Objective

---

- 1: Initialize base station  $\textcircled{S}$  with  $|K|$  RBGs, allocation policy  $\pi$  and threshold  $\kappa$
  - 2: **repeat**
  - 3:   Randomly generate  $\hat{N}_0$  initial users
  - 4:   **for**  $t = 1, \dots, T$  **do**
  - 5:     New users arrive at the  $\textcircled{S}$  according to the Poisson distribution
  - 6:     Allocating  $|K|$  RBGs to active users following the policy  $\pi$
  - 7:   **end for**
  - 8:   Calculate the 5%-tile user data rate  $\phi_T$
  - 9:   Update allocation policy  $\pi$  following specific method
  - 10: **until**  $\phi_T \geq \kappa$
- 

Solving this objective is mathematically intractable because of two main challenges. On the one hand, the actual user data rate is affected by multiple factors of different layers, it is almost impossible to conduct cross-layer optimization through traditional model-based methods. What's more, the 5TUDR is time-varying and not strictly increasing considering the bursty traffic, which confuses the optimization direction. On the other hand, allocating multi RBGs to multi users is a joint action, this produces exponential action space. Thus it is difficult to find the optimal solution through centralized optimization. In the following sections, we propose a stochastic game framework for the RBG allocation, and solve the problem via distributional method.

### III. STOCHASTIC GAME FRAMEWORK FOR RBG ALLOCATION

A traditional model-based scheduler allocates all the RBGs following the same allocation policy. Such operation allows the scheduler to stably optimize a fixed objective, e.g., system throughput or fairness. But the static policy can not adapt to the complex network conditions, sometimes it is even opposed to the original objective. To improve the flexibility and adaptability, we consider making allocation through distributional policy. Assume there are multi schedulers in one network, and each RBG is allocated by an independent scheduler. The Fig. 2 illustrates the detailed framework. Thus the RBG allocation process can be modeled a stochastic game, which is the generalization of the Markov Decision Process (MDP) for the multi-agent cases.

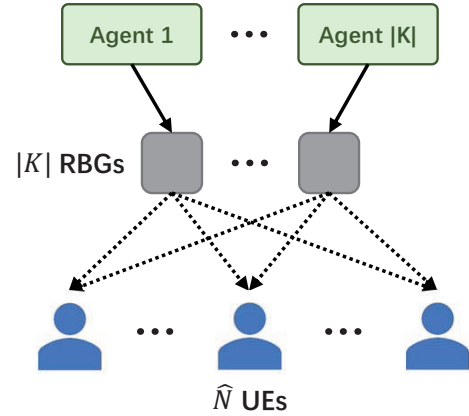


Fig. 2. RBG allocated by different agents.

**Definition 1.** This stochastic game can be defined by a tuple  $\mathcal{E} = \langle \mathcal{S}, \mathcal{U}, \mathcal{A}, \mathcal{P}, r, \mathcal{Z}, \mathcal{O}, \gamma \rangle$  [19], where:

- $\mathcal{S}$ : the global state space;
- $\mathcal{U}$ : the action space;
- $\mathcal{A}$ : a set of agents and each agent is identified by  $a \in \mathcal{A}$ ;
- $\mathcal{P}(s'|s, \mathbf{u})$ :  $\mathcal{S} \times \mathcal{U} \times \mathcal{S} \rightarrow [0, 1]$ , the transition probability, where  $\mathbf{u} \in \mathcal{U} \equiv \mathcal{U}^{|\mathcal{A}|}$ ;
- $r(s, \mathbf{u}, a)$ :  $\mathcal{S} \times \mathcal{U} \times \mathcal{A} \rightarrow \mathbb{R}$ , the reward function which specifies an agent its specific reward, where  $\mathbf{u} \in \mathcal{U} \equiv \mathcal{U}^{|\mathcal{A}|}$ ;
- $\mathcal{Z}$ : agents draw its individual observation  $z \in \mathcal{Z}$ ;
- $\mathcal{O}(s, a)$ :  $\mathcal{S} \times \mathcal{A} \rightarrow \mathcal{Z}$ , the observation function;
- $\gamma$ :  $\gamma \in [0, 1]$  is a discount factor.

We further consider the partially observable settings, in which agents get local observations  $o_a^t$  based on observation function  $\mathcal{O}(s, a)$  [20] [21]. Thus each agent has an action-observation history  $\tau^a \in \mathcal{T} \equiv (\mathcal{Z} \times \mathcal{U})^*$ , which is conditioned by a stochastic policy  $\pi^a(u^a | \tau^a) : \mathcal{T} \times \mathcal{U} \rightarrow [0, 1]$ . All the local policies constitutes the joint-policy  $\pi$ , and the distribution over the joint-action can be formulated as:

$$P(\mathbf{u} | s_t) = \prod_a \pi^a(u^a | \tau^a) \quad (7)$$

In the following sections, the specific forms of observations, actions in the stochastic game of RBG allocation are elaborated in detail.

#### A. Cross-layer Observation and Adaptive Action

Initialize a base station with a set of  $K$  RBGs, and randomly generate  $\hat{N}$  active users. There will be  $|K|$  agents responsible for the allocation based on its local observations of the network. For a traditional model-based scheduler, it usually observes the single-layer information of the network, such as MAC layer or PHY layer. Take PF scheduling for instance, it makes allocation by tracking the historical AUDR and estimated AUDR, which comes from the MAC layer. This is mainly because of the limitation of mathematical models.

As the allocation performance is affected by multi-layer information, the agents are expected to get cross-layer observations, so that it can make more comprehensive decisions. To handle the cross-layer network data, the Deep Neural Network (DNN) is introduced to serve as the backbone of the agents, which builds the model from the perspective of data rather than the original mathematical relationship [22].

TABLE I  
NETWORK DATA OF GLOBAL STATE

| Attr.                            | Layer | Dim.         |
|----------------------------------|-------|--------------|
| RSRP                             | PHY   | 1            |
| Average of RB CQI (each RBG)     | PHY   | Num. of RBGs |
| Buffer size                      | MAC   | 1            |
| Scheduled times                  | MAC   | 1            |
| Olla offset                      | MAC   | 1            |
| Historical user data rate (HUSR) | MAC   | 1            |

Table I illustrates the selected network data in this paper. Reference Signal Receiving Power (RSRP) is one of the key parameters that can represent the wireless signal strength in LTE networks. This metric is a constant through the interaction between user and base station, which can straightforwardly reflects the level of the final transmission rate. Channel Quality Indicator (CQI) is the information indicator of channel quality, which represents the current channel quality and corresponds to the signal-to-noise ratio (SNR) of the channel. For each user, it has different CQI on different RBGs. This metric is adopted referring to the OP scheduling. To maximize the throughput, OP scheduling prefers to allocate more RBGs to advantageous users who take higher expected transmission rate. Here the buffer size is the data packets need to be transferred. Regardless of the CQI, the buffer size can also serve as key decision information in scheduling. Assume there is an algorithm which prefers to allocate RBGs to users with low buffer size, this can also improve the transmission rate by reducing the transmission time of users. The schedule times here is designed for the probability fairness, which indicates whether all the users have same probability to get the RBGs. Finally, the historical AUDR is collected referring to the PF scheduling. The PF scheduling calculates the allocation priorities for each user by tracing the historical AUDR and estimated AUDR. And the configured agents are hoped to

take such metrics into consideration when conducting the allocation.

Denote by  $F = \{\mathbf{f}^1, \dots, \mathbf{f}^{|F|}\}$  the set of selected network data from different layers, then the global state can be formulated as a feature matrix:

$$\mathbf{S} = (\mathbf{f}^1, \dots, \mathbf{f}^{|F|}), \mathbf{f}^i = (f_1^i, \dots, f_{\hat{N}}^i)^T \quad (8)$$

For the *bursty traffic*, the feature matrix is a variable due to the time-varying number of active users in the base station. It also means the agents need to allocate fixed RBGs to an uncertain number of users. However, the Fully-Connected Neural Networks (FCNN) only accepts input with fixed dimensions along with the output [23]. To address this problem, the projection operation is applied to the state matrix  $\mathbf{S}$ :

$$\mathbf{s} = \mathfrak{F}(\mathbf{S}^T \mathbf{S}), \mathbf{s} \in \mathbb{R}^{|F|^2} \quad (9)$$

where  $\mathfrak{F}(\cdot)$  is a flattening operator, it flattens a matrix into a vector by row and generates a vector with fixed dimensions. Because many selected attributes are continuous values, which is almost impossible to generate absolutely same  $\mathbf{s}$ .

TABLE II  
NETWORK DATA OF LOCAL OBSERVATIONS

| Attr.                            | Layer | Dim. |
|----------------------------------|-------|------|
| RSRP                             | PHY   | 1    |
| Average of RB CQI                | PHY   | 1    |
| Buffer size                      | MAC   | 1    |
| Scheduled times                  | MAC   | 1    |
| Olla offset                      | MAC   | 1    |
| Historical user data rate (HUSR) | MAC   | 1    |

Based on the global state, we design the local observations using the subset of  $F$  (listed in Table II), which can be written as:

$$\mathbf{O} = (\mathbf{f}^1, \dots, \mathbf{f}^{|\hat{F}|}), \mathbf{f}^i = (f_1^i, \dots, f_{\hat{N}}^i)^T, \hat{F} \subset F \quad (10)$$

This allows all the agents share some public network data, such as the buffer size or the historical AUDR of the users, while keeping the unique attributes of RBG such as the Channel Quality Indicator (CQI) on each user. It also motivates the agents to explore diverse policies when maximizing its objective, because these unique attributes change flexibly in the allocation process.

Similarly, the projection operation is also applied to the local observations to generate a fixed vector:

$$\mathbf{o} = \mathfrak{F}(\mathbf{O}^T \mathbf{O}), \mathbf{o} \in \mathbb{R}^{|\hat{F}|^2} \quad (11)$$

As the DNN is leveraged to be the backbone of the agents, we let  $\pi_{\theta_a}^a$  the policy parameterized by a fully-connected network with parameters  $\theta_a$ . For RBG allocation, the policy accepts a local observation  $\mathbf{o}^a$  and generates  $\hat{N}$  priorities for  $\hat{N}$  users, then the RBG is allocated to the user with highest priority:

$$u_a = \arg \max \pi_{\theta_a}^a(\mathbf{o}^a) \quad (12)$$



All the local actions constitute the joint-action  $\mathbf{u}$ , which can be written as follows when there are  $|K|$  RBGs and  $\hat{N}$  active users in the base station:

$$\mathbf{u} = (u^1 \dots u^{|K|}), u^a \in \{1, \dots, \hat{N}\} \quad (13)$$

Similar to time-varying problem in the definition of observation, the policy also needs to make adaptive allocation due to the time-varying  $\hat{N}$ . However, limited by the properties of the FCNN, it can not generate adaptive output.

To address the problem, it is reasonable to assume there is a maximum number of active users, which can be denoted as  $M$ . Thus the users in the base station can be divided into two parts:  $\hat{N}$  active users and  $M - \hat{N}$  virtual users. Then set the output layer of the policy network with  $M$  neurons to generate  $M$  priorities. Finally, select the action using the former  $\hat{N}$  values. The Fig. 3 illustrates the operation.

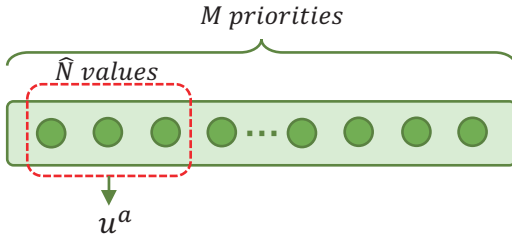


Fig. 3. Adaptive action

Moreover, all the network data of the virtual users will be set as 0, and this will not affect the final values of the observation vector. With a lot of interactions and training, the agents are feasible to learn the corresponding pattern.

### B. Fully Cooperative Game

As the final user data rate is determined by the usage of all resources, so we set all the agents are fully cooperative in this game, this implies that all the agents receive the same reward and make it as a global reward [17]:

$$r(\mathbf{s}, \mathbf{u}, a) = r(\mathbf{s}, \mathbf{u}, a'), \forall a, a'. \quad (14)$$

For achieving the highest 5TUDR, this reward function outputs the gain performance in 5TUDR for each state-action pair. The specific form of the reward function will be carefully discussed in the latter sections. As the agents are set to be fully cooperative, the reward function is redefined as  $r(\mathbf{s}, \mathbf{u})$  for convenience in the following sections. Considering the more general situation, the optimization objective in Algo. 1 is reformulated as:

$$\max_{\pi} V^{\pi}(\mathbf{s}) = \mathbb{E} \left[ \sum_{t=0}^{+\infty} \gamma^t r_t(\mathbf{s}_t, \mathbf{u}_t) | \pi, \mathbf{s}_0 = \mathbf{s} \right] \quad (15)$$

where  $\pi = \{\pi_{\theta_1}^1, \dots, \pi_{\theta_{|K|}}^{|K|}\}$  being the joint policy. Eq. 15 indicates the expected return if all the agents executing the joint policy  $\pi$  from the global state  $\mathbf{s}$ . As each agent makes allocation following its local observations, the cooperative relationship allows the agents to use diverse policies to maximize the collective objective. So the induced joint-policy is more flexible and adaptive.

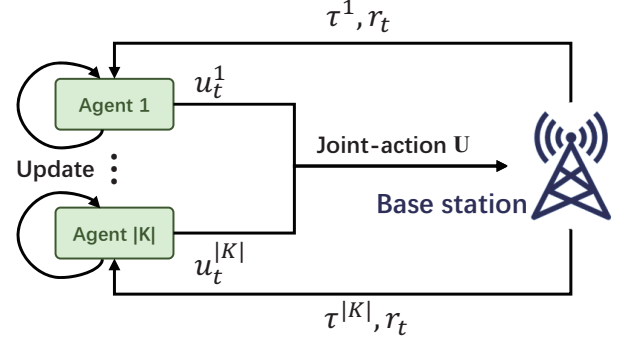


Fig. 4. MARL framework for RBG allocation

### C. Solution

So far the optimization objective is transformed into a multi-agent, fully cooperative game. For such a distributional optimization problem, the Nash Equilibrium (NE) is always used to describe the solution [17]. The NE assumes that the agents always make a best response in the game, but sometimes the agents can cooperate even though it is not the best response. Thus Halpern et al. propose a new equilibrium definition, which is entitled Perfect Cooperative Equilibrium (PCE) [24]. In the PCE, the payoff of the agent is at least high as in any NE, but it can help explain the cooperation observed in practice.

To elaborate the PCE in this game, let  $V^a(\pi)$  denote the expected utility of agent  $a$  if the joint-policy  $\pi = \{\pi^1, \dots, \pi^{|K|}\}$  is performed. Considering the cooperative relationship, the  $V^a(\pi)$  is assumed to be additive so that:

$$V^{\pi}(\mathbf{s}) = \sum_{a \in \mathcal{A}} V_a(\pi) \quad (16)$$

**Definition 2.** Given the game  $\mathcal{E}$ , let  $\tilde{V}^a(\pi)$  denote the best utility that agent  $a$  can obtain if the other agents  $\mathcal{A} - \{a\}$  make best response, then the joint-policy  $\pi$  is a perfect cooperative equilibrium (PCE) if for all  $a \in \mathcal{A}$ , it holds:

$$V^a(\pi) \geq \tilde{V}^a(\pi) \quad (17)$$

This implies that in a PCE, each agent acts at least as well as other agents if the other agents were best-responding. Therefore, each agent should make every effort to avoid falling behind. To find such a PCE, each agent is required to learn from the interactions with the environment following special learning method. In next section, an learning algorithm for maximizing the objective in Eq. 15 is proposed based on Multi-Agent Reinforcement Learning (MARL).

## IV. MARL-BASED ALGORITHM

In this section, we first give the basic elements of MARL according to the defined stochastic game. Then a QMIX learning algorithm is introduced to maximize the expected long-term return defined in Eq. 15. Finally, the detailed algorithm of MARL for RBG allocation is summarized.

### A. MARL framework for RBG Allocation

The fully cooperative multi-agent task considers learning from the Decentralized Partial Observable Markov Decision Process (Dec-POMDP) which has identical definition of Def. 1 [21]. The Fig. 4 illustrates the main framework of MARL for RBG allocation, where a set of  $K$  RBGs are allocated by  $|K|$  agents respectively. At each TTI, the base station generates local observations for each agent, and the agents make the corresponding actions. According to the global state and joint-action, the base station calculates a reward for the state-action pair. Finally, the agents update its policy via RL algorithms based on the experience which comprises observations, actions and rewards.

Furthermore, the state-value function and action-value function are introduced to characterize the expected return:

$$\begin{aligned} V^\pi(\mathbf{s}) &= \mathbb{E} \left[ \sum_{t=0}^{+\infty} \gamma^t r_t | \pi, \mathbf{s}_0 = \mathbf{s} \right] \\ Q^\pi(\mathbf{s}, \mathbf{u}) &= \mathbb{E} \left[ \sum_{t=0}^{\infty} \gamma^t r_t | \pi, \mathbf{s}_0 = \mathbf{s}, \mathbf{u}_0 = \mathbf{u} \right] \end{aligned} \quad (18)$$

The  $V^\pi(\mathbf{s})$  is identical to the optimization objective defined in Eq. 15. As for  $Q(\pi, \mathbf{s}, \mathbf{u})$ , this represents the expected return if the agents executing joint policy  $\pi$  after taking joint-action  $\mathbf{u}$  at global state  $\mathbf{s}$ . Moreover, the value-based RL algorithms choose the action at each step following:

$$\mathbf{u} = \arg \max_{\mathbf{u}} Q^\pi(\mathbf{s}, \mathbf{u}) \quad (19)$$

The Bellman Optimal Theorem indicates the relationship between the optimal policy and the action-value function, and the  $Q^*(\mathbf{s}, \mathbf{u})$  can be learnt via Q-learning in [25].

**Theorem 1.** *Let  $\Pi$  be the set of all non-stationary and randomized policies. Define:*

$$Q^*(\mathbf{s}, \mathbf{u}) := \sup_{\pi \in \Pi} Q^\pi(\mathbf{s}, \mathbf{u}) \quad (20)$$

*which is finite since  $Q^\pi(\mathbf{s}, \mathbf{u})$  is bounded between 0 and  $\frac{1}{1-\gamma}$ . There exists a stationary and deterministic policy  $\pi$  such that for all  $\mathbf{s} \in \mathcal{S}$  and  $\mathbf{u} \in \mathcal{A}$ ,*

$$Q^\pi(\mathbf{s}, \mathbf{u}) = Q^*(\mathbf{s}, \mathbf{u}) \quad (21)$$

*Such  $\pi$  is referred as an optimal policy.*

Considering high dimensional state space, it is essential to use function approximation as a compact representation of action values. Deep neural network provides powerful approximation abilities to undertake the task. Hornik et al. has proved that standard multilayer feedforward networks are capable of approximating any measurable function to any desired degree of accuracy [26]. Let  $Q_\theta^\pi(\mathbf{s}, \mathbf{u})$  denote the action-value function parametrized by a DNN with parameters  $\theta$ . Thus the optimization objective is to find  $\theta$  that holds  $Q_\theta^\pi(\mathbf{s}, \mathbf{u}) \approx Q^*(\mathbf{s}, \mathbf{u})$ . And the optimal parameters  $\theta$  can be approached by minimizing the following temporal difference error:

$$\mathcal{L}(\theta) = \left[ r + \gamma \max_{\mathbf{u}'} Q_\theta^\pi(\mathbf{s}', \mathbf{u}') - Q_\theta^\pi(\mathbf{s}, \mathbf{u}) \right]^2 \quad (22)$$

### B. QMIX Based RBG allocation

The partially observable setting let each agent maintain an observation history  $\tau^a \in \mathcal{T}$ . For the environment, there is a joint action-observation history  $\tau \in \mathcal{T}^{|A|}$ . Then the action-value function is rewritten as  $Q_\theta^\pi(\tau, \mathbf{u})$ . It is difficult to straightforwardly learn the optimal action-value function due to the exponential joint-action space. Moreover, the  $Q_\theta^\pi(\tau, \mathbf{u})$  can not be used to apply decentralized control. In order to address this problem, the QMIX is introduced to learn the joint-action value function via value decomposition. For convenience, we refer  $Q_{tot}$  to the  $Q_\theta^\pi(\tau, \mathbf{u})$  in the following sections.

The QMIX is a model-free, value-based, off-policy, decentralized execution and centralized training algorithm towards cooperative task, which takes DNN function approximator to estimate the action-value function [27]. QMIX allows each agent  $a$  maintain an individual action-value function  $Q_a(\tau^a, u^a)$ , which conditions on action-observation history  $\tau^a$  and local action  $u^a$ . Thus the decentralized execution is tractable based on the individual action-value functions. For centralized training, the QMIX leverages a mixing network to calculate the centralized action-value function  $Q_{tot}$ . What's more, the QMIX ensures that the global argmax operation applied on the centralized  $Q_{tot}$  generates the same result as a set of argmax operation applied on each  $Q_a$ :

$$\arg \max_{\mathbf{u}} Q_{tot} = \left( \begin{array}{c} \arg \max_{u^1} Q_1(\tau^1, u^1) \\ \vdots \\ \arg \max_{u^n} Q_n(\tau^n, u^n) \end{array} \right) \quad (23)$$

Mathematically, this imposes a monotonicity constraint on the relationship between  $Q_{tot}$  and each  $Q_a$ :

$$\frac{\partial Q_{tot}}{\partial Q_a} \geq 0, \forall a. \quad (24)$$

Since the QMIX is designed for the fully cooperative task, this constraint guarantees that the performance gain of each agent will contribute to the gain of global return. Moreover, it allows the agents to freely explore all the possible solutions, as long as it brings the increase of action-value function. Fig. 5 illustrates the framework of QMIX. And the detailed workflow of the QMIX can be found in [29].

### C. Distributional Joint-Policy and Centralized Joint-Policy

In QMIX, there are multi Q-value networks which represent the policies agents respectively. Consider the fully cooperative task, there are two kinds of feasible joint-policy can be learned in practice—distributional joint-policy and centralized joint-policy.

For the distributional joint-policy, we conduct the training process using multi networks for the agents. This configuration allows all the agents to explore different policies, while updating its parameters independently. All the agents using unique policies to optimize the collective objective together, which induces a distributional joint-policy. However, as the centralized action-value function is composed of the individual action-value functions, which means more agents produce more variance. What's more, the increasing variance probably

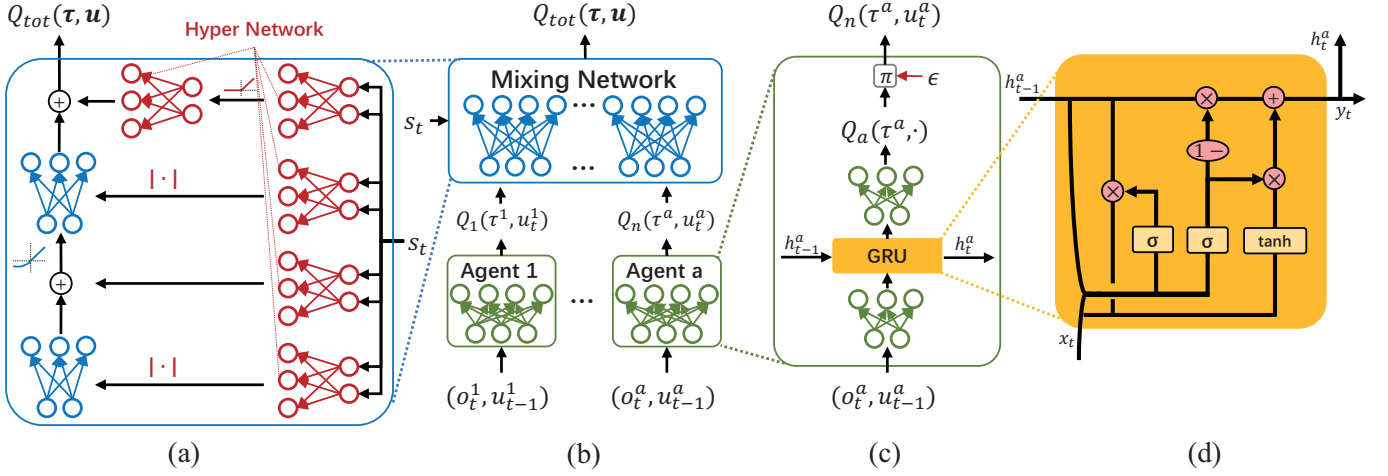


Fig. 5. (a) Mixing network structure. In red are the hypernetworks that produce the weights and biases for mixing network layers shown in blue. (b) The overall QMIX architecture. (c) Agent network structure. (d) The GRU structure, where  $h$  is the hidden information used for recursion [28].

affects the convergence of the training process, so as to cause the collapse of the model. But the distributional policy maintains high flexibility and adaptability, which is more effectively when handling the complex conditions. The Algo. 2 first illustrates the distributional version of the MARL-based algorithm for RBG allocation.

As for the centralized joint-policy, we conduct the training process only using single network for the agents. In other words, all the agents can share the network parameters, because they accept the observations with same definition and structure in the game. The learned joint-policy acts as same as the traditional schedulers, which allocates the RBGs one by one using same policy. This will not destroy the cooperative game relationship between the agents, because the final performance is consistent with the quality of the policy. Compared with the distributional version discussed before, it is more easily to train the single network. Accordingly, the flexibility and adaptability will be greatly affected. The Algo. 3 illustrates the centralized version of the MARL-based algorithm for RBG allocation.

## V. REWARD FUNCTION CONFIGURATION

For the cooperative game, there is a global reward function which characterizes the collective optimization objective. The definition and measurement of the reward function straightforwardly affects the exploration direction and the final preference of the learned policy. However, the design of the reward function lacks rigor theoretical support in practice, most of the configurations are always experience-based. In this paper, the learning system is expected to maximize the 5TUDR of the communication network. In this section, we dive into the design of reward function, and discuss the taken insights from comprehensive perspective.

### A. Consistency

Firstly, the reward function should be consistent with the optimization objective. For the 5TUDR, it is selected from a series of individual user data rate. Regardless of the special

selection method, the essence of the 5TUDR is user data rate. Thus the reward function should be a mapping from the user data rate to the real number set.

Intuitively, to maximize the 5TUDR, it is natural to take the increment of 5TUDR as the reward function. This indicates that each action of agents should contribute to the 5TUDR, but this reward function is founded to be incompatible with the bursty traffic. To demonstrate the conflict, we trace the variation of 5TUDR in one random simulation which lasts 500TTIs. As can be seen in the Fig. 6, the increment of 5TUDR is always 0, and there are some sharp increase and sharp decrease in the figure. The sharp decrease is because of the arrival of new users, if a new user arrives at the base station at TTI  $t$ , the  $\Phi_t$  will suddenly become 0 because the new user has no transferred data packets. Once the new user is scheduled, its user data rate will soar, which causes the sharp increase. Therefore, this reward function can not cope with the bursty traffic and properly represent whether the actions are good or bad. Moreover, this configuration can not provide enough motivations for the agents to focus on the inferior users because of the confusing reward.

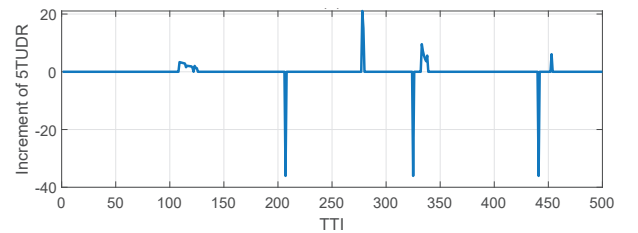


Fig. 6. The increment of 5TUDR between two adjacent TTI in one simulation.

Similarly, we can take the increment of AUDR as the reward function, which aims to improve the 5TUDR via improving the AUDR. But this will also be affected by the time-varying number of the active users. Fig. 7 illustrates the increment of AUDR between two adjacent TTI in one simulation. There are also some sharp increase and sharp decrease in the figure. Here the sharp decrease is mainly because the arrival of



**Algorithm 2** QMIX for RBG Allocation (Distributional)

- 1: Initialize  $|K|$  agent networks  $Q_{\theta_1} : \mathcal{T} \rightarrow \mathbb{R}^{|\mathcal{U}|}, \dots, Q_{\theta_{|K|}} : \mathcal{T} \rightarrow \mathbb{R}^{|\mathcal{U}|}$  with parameters  $\theta = (\theta_1, \dots, \theta_{|K|})$ .
- 2: Initialize hyper-network  $\mathcal{H}_\eta$  and mixing network  $\mathcal{M}_{\beta_0}$  with parameters  $\eta$  and  $\beta_0$ .
- 3: Set a replay buffer  $\mathcal{B}$ , learning rate  $\lambda$ , discount factor  $\gamma$  and  $\epsilon$ .
- 4: **for** Epoch = 1, ...,  $E$  **do**
- 5:   Initialize the base station  $\textcircled{S}$  with a set of  $K$  RBGs and randomly generate  $\hat{N}_1$  initial users.
- 6:   Each agent receives the initial local observation  $\mathbf{o}_1^a$
- 7:   **for**  $t = 1, \dots, T$  **do**
- 8:     New users arrive at the  $\textcircled{S}$  according to the Poisson distribution
- 9:     Each agent  $a$  makes local action  $u_t^a$  following the  $\epsilon$ -greedy policy and observe a new local state  $\mathbf{o}_{t+1}^a$ .
- 10:    Execute the joint-action  $\mathbf{u}_t$  and observe the global reward  $r_t$ , then observe a new global state  $\mathbf{s}_{t+1}$
- 11:    Store the following transition in  $\mathcal{B}$ :
 
$$(\mathbf{s}_t, r_t, \mathbf{s}_{t+1}, \mathbf{o}_t^a, \mathbf{o}_{t+1}^a, \mathbf{u}_{t-1}^a, \mathbf{u}_t^a, \mathbf{h}_{t-1}^a, \mathbf{h}_t^a)$$
- 12:    Sample a random minibatch of  $b$  transitions from  $\mathcal{B}$ :
 
$$(\mathbf{s}_i, r_i, \mathbf{s}_{i+1}, \mathbf{o}_i^a, \mathbf{o}_{i+1}^a, \mathbf{u}_{i-1}^a, \mathbf{u}_i^a, \mathbf{h}_{i-1}^a, \mathbf{h}_i^a)$$
- 13:    Get  $|K|$  individual  $Q$  values:
 
$$\mathbf{Q}_{\theta_a} = (Q_{\theta_1}(\tau^1, u_i^1), \dots, Q_{\theta_{|K|}}(\tau^{|K|}, u_i^{|K|}))$$

$$\mathbf{Q}'_{\theta_a} = (Q_{\theta_1}(\tau^{1'}, u_{i+1}^1), \dots, Q_{\theta_{|K|}}(\tau^{|K|'}, u_{i+1}^{|K|}))$$
- 14:    Calculate weights for mixing network, then get  $Q_{tot}$  and  $Q'_{tot}$ :
 
$$\beta_i \leftarrow \mathcal{H}_\eta(\mathbf{s}_i), \beta_{i+1} \leftarrow \mathcal{H}_\eta(\mathbf{s}_{i+1})$$

$$Q_{tot}(\tau, \mathbf{u}_i, \mathbf{s}_i; \beta_i) \leftarrow \mathcal{M}_{\beta_i}(\mathbf{Q}_{\theta_a})$$

$$Q'_{tot}(\tau', \mathbf{u}_{i+1}, \mathbf{s}_{i+1}; \beta_{i+1}) \leftarrow \mathcal{M}_{\beta_{i+1}}(\mathbf{Q}'_{\theta_a})$$
- 15:    Set  $y_i = r_i + \gamma \max_{\mathbf{u}_{i+1}} Q'_{tot}(\tau', \mathbf{u}_{i+1}, \mathbf{s}_{i+1}; \beta_{i+1})$
- 16:    Update all the agent networks and hyper-network by minimizing the following loss:

$$\mathcal{L}(\theta, \eta) = \sum_{i=1}^b \left[ (y_i - Q_{tot}(\tau, \mathbf{u}_i, \mathbf{s}_i; \beta_i))^2 \right]$$

- 17:   **end for**
- 18: **end for**

new users, which reduces the instantaneous average value. In particular, the sharp increase is mainly caused by "lucky" actions. The "lucky" means that the agents coincidentally allocate the RBGs to the user with very high CQI values. Thus the corresponding high reward is not because the agents takes good actions, it is just due to the good luck. Moreover, the agents will be severely confused by the CQI values, which is more easier to get high reward as long as it allocates all the RBGs to users with highest CQI values. Not only that, the CQI values will always make great difference if the reward function conditions on the user data rate. Because the CQI values varies frequently and greatly affects the transmission

**Algorithm 3** QMIX for RBG Allocation (Centralized)

- 1: Initialize  $|K|$  agent networks  $Q_{\theta_1} : \mathcal{T} \rightarrow \mathbb{R}^{|\mathcal{U}|}, \dots, Q_{\theta_{|K|}} : \mathcal{T} \rightarrow \mathbb{R}^{|\mathcal{U}|}$  with parameters  $\theta = (\theta_1, \dots, \theta_{|K|}), \theta_1 = \dots = \theta_{|K|}$ .
- 2: Execute step 2-18 in Algo. 2

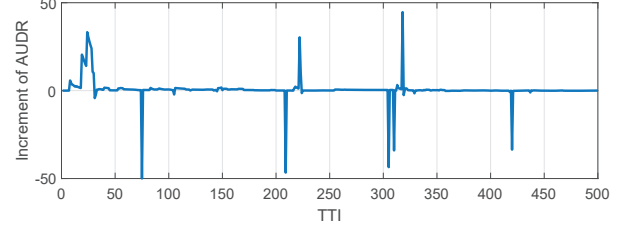


Fig. 7. The increment of AUDR between two adjacent TTI in one simulation.

rate, but it can not be controlled by the agents or the base station.

**B. Boundedness**

Secondly, we discuss the necessity of the boundedness. As the user data rate is utilized to calculate the reward, it is inevitable to suffer from the "lucky" actions. On the one hand, the "lucky" actions promote the usage of the network resources, which produces high system throughput. On the other hand, the "lucky" actions confuse the fair evaluation of the other actions. Thus the reward function is expected to be bounded, it guarantees the "luck" actions will not be over rewarded while keeping its positive influence. Moreover, a normalized and bounded reward function will contribute to the convergence of the training process. [11]

**C. Adopted reward function**

Based on the former discussions, we finally define the reward function as below:

$$h(x) = \frac{1}{1 + e^{-x}}$$

$$r_t = -h\left(\sum_{i=1}^{N_t} \psi_t^i - \sum_{i=1}^{N_{t-1}} \psi_{t-1}^i\right), \psi_0^i = 0 \quad (25)$$

where  $h(x)$  is the sigmoid function and  $N$  is the number of total arrived users in the base station. And the reward function of Eq. 25 is based on three insights:

- This reward function conditions on the sum user data rate, which has nothing to do with the time-varying number  $N$  of total arrived users.
- As analyzed before, the CQI values always attracts the attention of agents as long as the reward function conditions on the user data rate. To reduce the influence of the CQI values, we requires the agents to decrease the increment of sum user data rate. Once the agents allocate most RBGs to the users with high CQI values, it leads to a sharp increase of sum user data rate but a smaller negative reward. That forces the agents to find

alternative solutions, which probably pay more attention to other attributes like historical AUDR and so on. In a word, this configuration explores more solutions to optimize 5TUDR at the cost of reducing a certain amount of AUDR.

- This reward function is bounded by using the sigmoid function  $h(x)$ , which normalizes the rewards of all possible actions. So even if some of the actions severely reduced the sum user data rate, it will not be over rewarded. Similarly, even if some of the actions greatly improve the sum user data rate, it will not be over punished. Such settings not only retain the dependence on CQI, but also increase the exploration of fairness.

\*\*\*\*\*

## VI. SIMULATION RESULTS AND NUMERICAL ANALYSIS

In this section, we demonstrate the simulation results and the corresponding numerical analysis. All the simulations are accomplished with the base station defined in Section II, and Table III illustrates the experiment parameters.

TABLE III  
PARAMETERS FOR BASE STATION.

| Parameters                       | Values      |
|----------------------------------|-------------|
| Transmit power for each RB       | 18 dBm      |
| Number of RB for each RBG        | 3           |
| Frequency bandwidth for each RBG | 10MHz       |
| Noise power density              | -174 dBm/Hz |
| Minimum MCS                      | 1           |
| Maximum MCS                      | 29          |
| Number of HARQ                   | 8           |
| Feedback period of HARQ          | 8           |
| Initial RB CQI value             | 4           |

### A. Model training

The agents are trained following the parameters listed in Table IV. Here one epoch represents one simulation which lasts 1000TTIs. Since the QMIX conducts off-policy training, thus we set the capacity of replay buffer bigger than the duration of one epoch, so that the agents can learn from the cross-epoch experiences. The batches means that the agents sample 10 minibatches whose size is 256 to apply training in each step. Moreover, a learning decay is introduced to optimize the training, which prevents the training from missing the better solution.

We conduct the training which sets different number of RBGs, and more RBGs means there will be more agents used for the allocation. Fig. 8 illustrates the variation of episode reward.

### B. Performance comparison

Considering the PF scheduling defined in Section II, the scheduler is most fairness-oriented when  $\alpha_1 = 0, \alpha_2 = 1$ . But the realistic situation might be different in the bursty traffic, because the scheduler may not attain the whole capacity of the network. To investigate the highest 5TUDR that the PF scheduling can achieve, we conduct a large amount

TABLE IV  
PARAMETERS FOR TRAINING.

| Parameters                      | Values   |
|---------------------------------|----------|
| Epochs                          | 100      |
| Duration of one epoch           | 1000TTIs |
| Learning rate                   | 1e-3     |
| Learning rate decay             | 1e-7     |
| Batches                         | 10       |
| Batch size                      | 256      |
| Replayer capacity               | 2000     |
| $\epsilon$                      | 1e-2     |
| Initial number of users         | 5        |
| Maximum number of users         | 10       |
| Average of Poisson distribution | 1e-2     |

of simulations with different schedulers which are listed in Table V. There are multiple PF schedulers that take different coefficients, and the RRF scheduling and the OP scheduling are set as reference.

TABLE V  
LIST OF SCHEDULERS.

| Schedulers     | Remarks  |
|----------------|--|
| RRF scheduling | N/A  |
| PF scheduling  | $\alpha_1 = 0, 0.02, \dots, 0.98, 1, \alpha_2 = 1$ |
| OP scheduling  | N/A  |

As can be seen in Fig. 9, the PF scheduling achieves higher 5TUDR while the  $\alpha_1$  increases, and it finally stabilized at about  $\alpha_1 = 0.5$ . Therefore, the following evaluations are performed with four schedulers, which are MARL scheduling, PF1 scheduling ( $\alpha_1 = 0, \alpha_2 = 1$ ), PF2 scheduling ( $\alpha_1 = 0.5, \alpha_2 = 1$ ) and RRF scheduling respectively.

Similar to the model training, 100 random simulations are performed to verify the performance of four schedulers, and the AUDR and 5TUDR are calculated respectively. Fig. 10 and Fig. 11 illustrate the corresponding cumulative distribution function of performance difference. As can be seen from the figure, the MARL scheduling outperforms the other three schedulers in most of the simulations considering the 5TUDR. Table VI illustrates the numerical performance comparison both in AUDR and 5TUDR.

TABLE VI  
PERFORMANCE DIFFERENCE (BITS/TTI)

|          | 3RBGs  |       | 5RBGs  |       | 7RBGs  |       |
|----------|--------|-------|--------|-------|--------|-------|
|          | AUDR   | 5TUDR | AUDR   | 5TUDR | AUDR   | 5TUDR |
| MARL-PF1 | 73.22  | 34.86 | 80.18  | 65.66 | 93.80  | 57.03 |
| MARL-PF2 | -42.65 | 18.09 | -47.03 | 44.62 | -68.41 | 37.49 |
| MARL-RRF | 132.68 | 24.72 | 135.42 | 61.71 | 96.11  | 31.72 |

### C. Policy analysis

The simulation results show that MARL scheduling can significantly improve the fairness while maintaining good performance in AUDR. To demonstrate the scheduling policy of the four schedulers, their allocation logs are deeply analyzed.

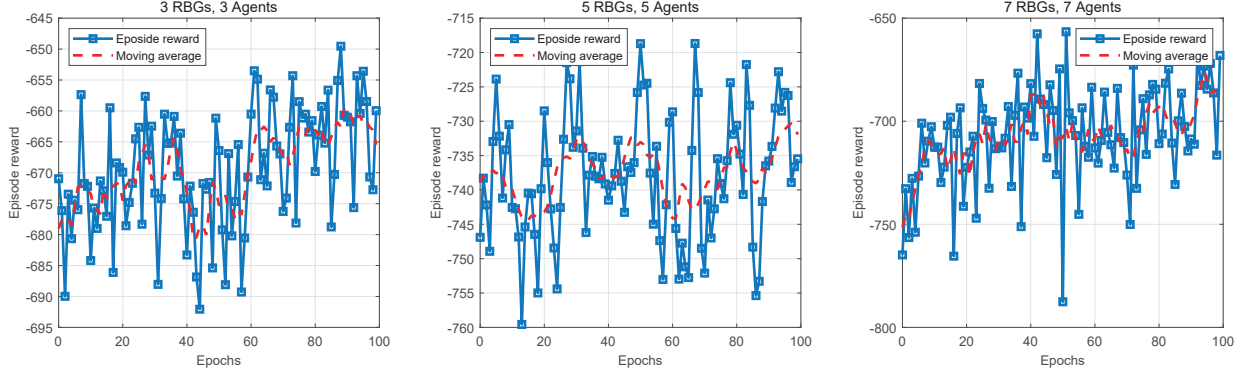


Fig. 8. Episode reward for epoch

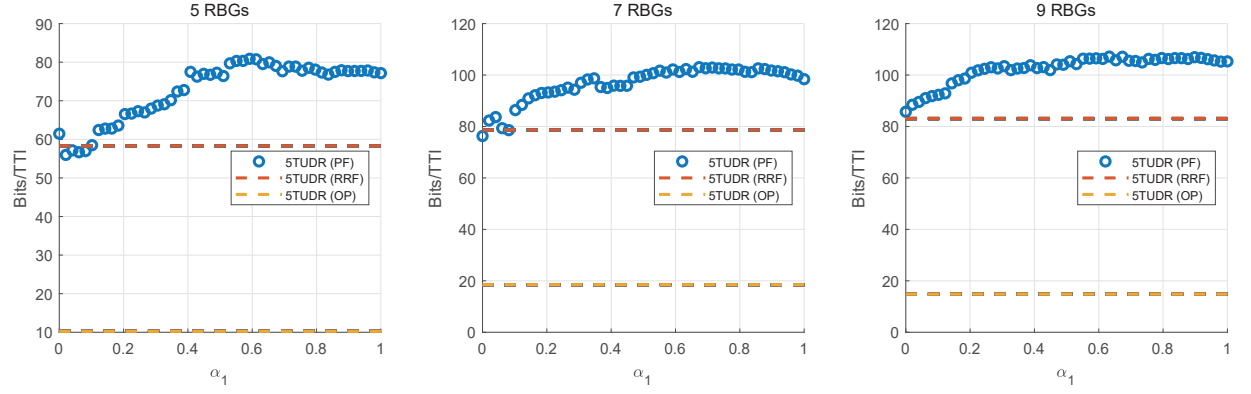


Fig. 9. Average 5TUDR of different schedulers in 1000 random simulations.

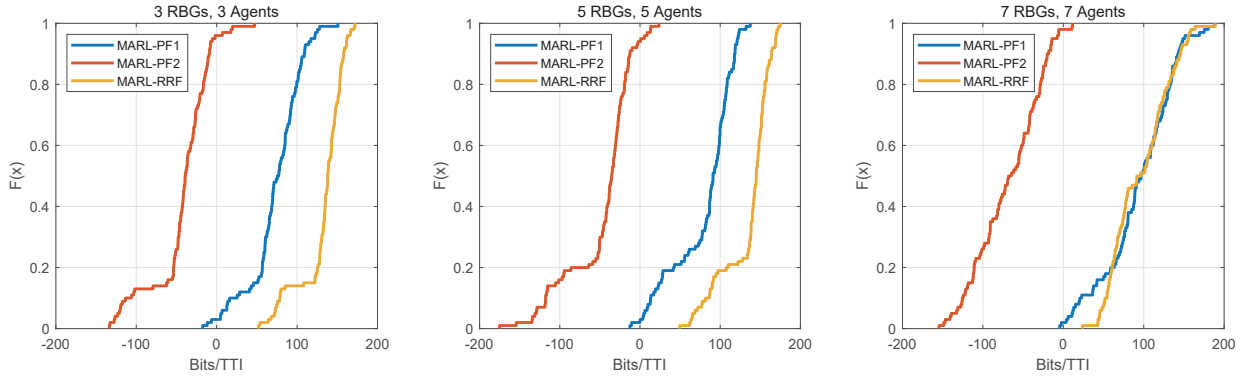


Fig. 10. CDF of performance difference in AUDR

Take the simulation with 3 RBGs for instance, there are two allocation examples listed in Table VII and Table VIII.

Table VII illustrates the allocation result in TTI=1. "T" and "F" represent the allocation result of RBGs. For instance, the "T" of 8th row means the RBG#1 is allocated to the UE#1 in Table VII. In this TTI, all the schedulers accept same users with specific attributes. Thus they make decisions based on same situations, which is more representative for the analysis of scheduling policy. For PF scheduling and RRF scheduling, they allocate all the RBGs to the UE#1, while the MARL distributes the resources into different users. Moreover, these scheduled users have smaller buffer size than the others. Since

each RBG is allocated by a specific agent, this example implies that the agents have different local policies. And an ensemble policy is built with these local policies.

Table VIII illustrates the allocation result in TTI=100, where "N/A" implies that the user has left base station. After a period of scheduling, the number of users changes and some users have left the base station. For the MARL scheduling, it insists following the distributional policy and allocating the RBGs to different users. But PF scheduling and RRF scheduling still allocate all the RBGs to a single user. Moreover, it is easy to find that users with lower buffer size take higher priorities in the two examples of MARL scheduling. Considering the good

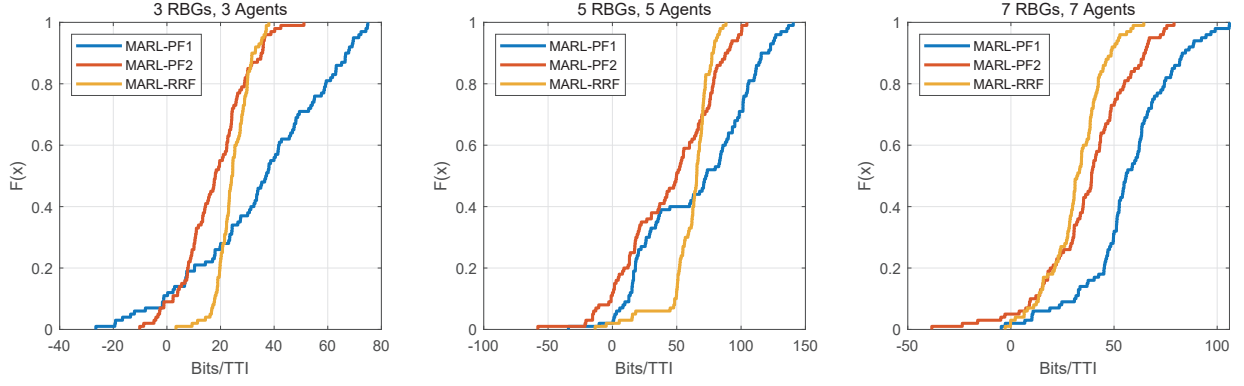


Fig. 11. CDF of performance difference in 5TUDR

TABLE VII  
ALLOCATION EXAMPLE 1 (TTI=1)

|      | Attr.  | UE#1  | UE#2  | UE#3  | UE#4 | UE#5  |
|------|--------|-------|-------|-------|------|-------|
| MARL | RSRP   | -99   | -90   | -96   | -88  | -70   |
|      | Buffer | 94768 | 96288 | 15528 | 4400 | 36400 |
|      | HUDR   | 0.0   | 0.0   | 0.0   | 0.0  | 0.0   |
|      | CQI#1  | 4     | 4     | 4     | 4    | 4     |
|      | CQI#2  | 4     | 4     | 4     | 4    | 4     |
|      | CQI#3  | 4     | 4     | 4     | 4    | 4     |
|      | RBG#1  | F     | F     | F     | F    | T     |
| PF1  | RBG#2  | F     | F     | F     | T    | F     |
|      | RBG#3  | F     | F     | F     | F    | F     |
| PF2  | RBG#1  | T     | F     | F     | F    | F     |
|      | RBG#2  | T     | F     | F     | F    | F     |
|      | RBG#3  | T     | F     | F     | F    | F     |
| RRF  | RBG#1  | T     | F     | F     | F    | F     |
|      | RBG#2  | T     | F     | F     | F    | F     |
|      | RBG#3  | T     | F     | F     | F    | F     |

performance in AUDR of MARL scheduling, we assume that the MARL scheduling allocate higher priorities to the users with lower buffer size, so as to minimize the total residence time of arrived users in the base station. This operation can maintain the performance in AUDR, while significantly improves the 5TUDR. To test the assumption, we calculate the residence time of all arrived users, which is shown in the Fig. 12.

To improve the 5TUDR, it is inevitable to decrease the AUDR. This figure indicates that MARL scheduling effectively decreases the residence time of arrived users, so as to meet the demand of users as fast as possible. So the MARL scheduling can significantly maintain the AUDR by improving the user data rate of users who have smaller buffer size. This can be also proved by the PF2 scheduling, as its total residence time is much less than the PF1 scheduling.

Except for the residence time, we investigate the allocation fairness of the three algorithms from the perspective of transmission time. This metric is the time that the arrived users get at least one RBG, which indicates that whether all the users

TABLE VIII  
ALLOCATION EXAMPLE 2 (TTI=100)

|      | Attr.  | UE#1  | UE#2  | UE#3  | UE#4 | UE#5  | UE#6  | UE#7 |
|------|--------|-------|-------|-------|------|-------|-------|------|
| MARL | RSRP   | -99   | -90   | -96   | N/A  | N/A   | N/A   | N/A  |
|      | Buffer | 91508 | 95910 | 6348  | N/A  | N/A   | N/A   | N/A  |
|      | HUDR   | 30.0  | 1.0   | 70.0  | N/A  | N/A   | N/A   | N/A  |
|      | CQI#1  | 7     | 9     | 7     | N/A  | N/A   | N/A   | N/A  |
|      | CQI#2  | 7     | 9     | 7     | N/A  | N/A   | N/A   | N/A  |
|      | CQI#3  | 3     | 14    | 9     | N/A  | N/A   | N/A   | N/A  |
|      | RBG#1  | T     | F     | F     | F    | F     | F     | F    |
| PF1  | RBG#2  | F     | F     | T     | F    | F     | F     | F    |
|      | RBG#3  | F     | F     | T     | F    | F     | F     | F    |
| PF2  | RSRP   | -99   | -90   | -96   | N/A  | -70   | -99   | N/A  |
|      | Buffer | 93152 | 83045 | 5091  | N/A  | 7815  | 5298  | N/A  |
|      | HUDR   | 100.0 | 200.0 | 80.0  | N/A  | 700.0 | 100.0 | N/A  |
|      | CQI#1  | 7     | 9     | 7     | N/A  | 25    | 7     | N/A  |
|      | CQI#2  | 7     | 9     | 7     | N/A  | 25    | 7     | N/A  |
|      | CQI#3  | 3     | 14    | 9     | N/A  | 27    | 5     | N/A  |
|      | RBG#1  | T     | F     | F     | F    | F     | F     | F    |
| RRF  | RBG#2  | T     | F     | F     | F    | F     | F     | F    |
|      | RBG#3  | T     | F     | F     | F    | F     | F     | F    |
| PF1  | RSRP   | -99   | -90   | -96   | N/A  | N/A   | -99   | N/A  |
|      | Buffer | 89327 | 81705 | 5211  | N/A  | N/A   | 7203  | N/A  |
|      | HUDR   | 40.0  | 200.0 | 100.0 | N/A  | N/A   | 200.0 | N/A  |
|      | CQI#1  | 7     | 9     | 7     | N/A  | N/A   | 7     | N/A  |
|      | CQI#2  | 7     | 9     | 7     | N/A  | N/A   | 7     | N/A  |
|      | CQI#3  | 3     | 14    | 9     | N/A  | N/A   | 5     | N/A  |
|      | RBG#1  | T     | F     | F     | F    | F     | F     | F    |
| PF2  | RBG#2  | T     | F     | F     | F    | F     | F     | F    |
|      | RBG#3  | T     | F     | F     | F    | F     | F     | F    |
| RRF  | RSRP   | -99   | -90   | -96   | N/A  | N/A   | -99   | N/A  |
|      | Buffer | 90175 | 86398 | 8594  | N/A  | N/A   | 5879  | N/A  |
|      | HUDR   | 40.0  | 90.0  | 70.0  | N/A  | N/A   | 50.0  | N/A  |
|      | CQI#1  | 7     | 9     | 7     | N/A  | N/A   | 7     | N/A  |
|      | CQI#2  | 7     | 9     | 7     | N/A  | N/A   | 7     | N/A  |
|      | CQI#3  | 3     | 14    | 9     | N/A  | N/A   | 5     | N/A  |
|      | RBG#1  | F     | F     | T     | F    | F     | F     | F    |
| PF1  | RBG#2  | F     | F     | T     | F    | F     | F     | F    |
|      | RBG#3  | F     | F     | T     | F    | F     | F     | F    |

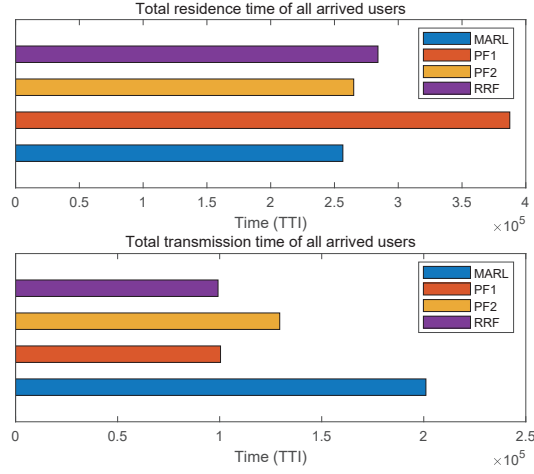


Fig. 12. Total residence time and total transmission time

have same opportunity to get resources. The statistical data is illustrated in Fig. 13. Here the "one" means that the scheduler allocates one RBG to a single user in TTI  $t$ . Similarly, "two" means that the scheduler allocates two RBGs to a single user in TTI  $t$ . It is obvious that the MARL scheduling attempts to let more user get RBGs in each TTI, so does the PF2 scheduling. This operation significantly increases the transmission time for all users, which leads to higher allocation fairness.

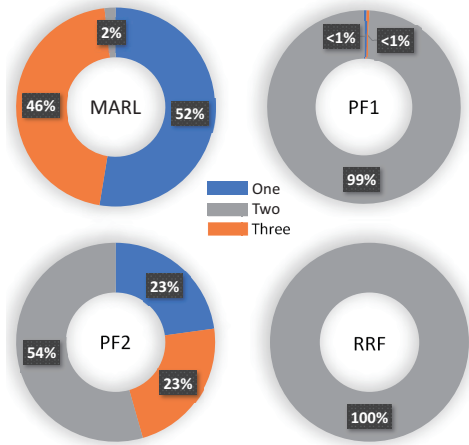


Fig. 13. Total transmission time with different number of RBGs

The utilization of cross-layer network data plays an important role in the allocation. By calculating the coefficient matrix between allocation results and selected network data, we can describe the scheduling policies in more detail. For MARL scheduling, each RBG is allocated by an independent agent. Since the agents are fully cooperative, it can take diverse methods to maximize the optimization objective. This probably produces an ensemble policy, which is more powerful to apply the global optimization for the RBG allocation. Fig. 14 implies that the three agents have independent scheduling policies, which is reflected by the different order of feature importance. For agent 1, it almost only focuses on the historical AUDR, and ignores the other attributes. For the agent 2, it mainly focuses on the historical user data rate, but also pays enough

attention to the RSRP and CQI. For the agent 3, the order of feature importance is more balanced, all the attributes provide a comprehensive reference when making allocations.

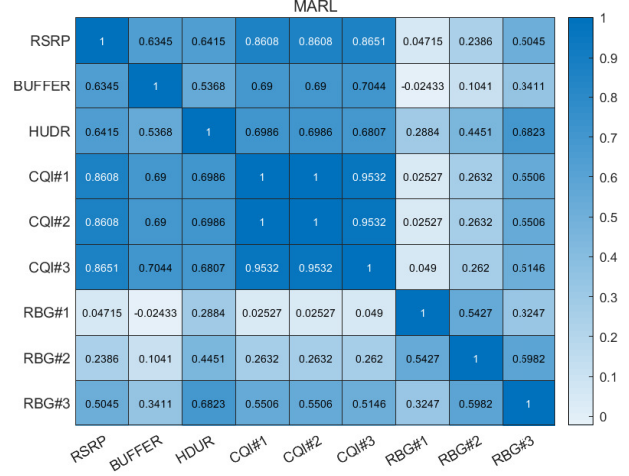


Fig. 14. Coefficient matrix of MARL scheduling

Note that the CQI doesn't get the highest feature importance for all the agents. This phenomenon is consistent with the design of reward function, which aims to depress the influence of CQI values to user data rate.

Finally, the scheduling policy of MARL scheduling can be summarized as:

- An ensemble and distributional scheduling policy composed of local policies of each agent. To maximize the 5TUDR, all the agents take different methods to maximize the collective object.
- Minimize the residence time of all arrived users to maintain the performance in AUDR. This can be seen in the former allocation example, where the MARL scheduler always allocates RBGs to users with lower buffer size. So it can meet the demand of more users in settled time and get higher AUDR.
- Provide more opportunity for users to get RBG, so as to improve the allocation fairness and promote the 5TUDR. It is obvious that the users can get resources more easily in the allocation of the MARL scheduling.

The simulations results and numerical analysis demonstrates that the MARL scheduler takes a more flexible scheduling policy, which effectively optimize the fairness of network while maintaining good performance in AUDR.

## VII. CONCLUSION

In this work, we investigate the problem scheduling, especially the RBG allocation. To maximize the fairness performance of the communication system, a fairness-oriented scheduler is proposed based on multi-agent reinforcement learning. This scheduler can allocates multiple RBGs to multiple users in the bursty, and achieve the highest 5%-tile user data rate compared with the PF scheduling and the RRF scheduling. Furthermore, we analyze the scheduling policy of the MARL scheduler is fully from diverse perspectives,



such as allocation time and so on. There are still much development space about the combination of user scheduling and reinforcement learning, and we will further improve the MARL-based scheduler which can optimize both the fairness and throughput. And we fully believe that the reinforcement learning will be utilized to make more pioneering work in the future.

## VIII. APPENDIX

### A. Benchmark schemes

Here gives the definition of PF scheduling and RRF scheduling which adopted in this work.

1) *Proportional Fairness Scheduling (PF)*: In the PF scheduling, any user with a non-empty buffer is a candidate to each RBG, and each RBG independently sort all users according to their PF values. Denote the PF value of user  $n$  on RBG  $k$  within transmission time interval (TTI)  $i$  by  $\beta_{n,k}[i]$ , then we have

$$\beta_{n,k}[i] = \frac{(R_{n,k}[i])^{\alpha_1}}{(\hat{T}_n[i])^{\alpha_2}}, \quad (26)$$

where  $\hat{T}_n[i]$  is the user's moving average throughput which can be expressed as

$$\hat{T}_n[i] = (1 - \gamma) \hat{T}_n[i - 1] + \gamma T_n[i - 1]. \quad (27)$$

In Eq. (27),  $\gamma$  is the moving average coefficient, and  $T_n[i - 1]$  is the actual TB size of user  $n$  in TTI  $i - 1$ . The users with higher PF value hold higher priority to occupy the RBG, and the PF value of a user to different RBGs can be different. Specifically, the PF scheduling can be expressed as

$$P_{PF}^*(k) = \arg \max_n \beta_{n,k}, \quad (28)$$

2) *Opportunistic Scheduling (OP)*: As the optimal scheduling in terms of systematic throughput maximization, opportunistic user scheduling is quite straightforward, which can be expressed as

$$P_{OP}^*(k) = \arg \max_n R_{n,k}. \quad (29)$$

It means each RBG selects the user suggesting the highest estimated data rate, which also implies the optimal scheduling in achieving the highest AUDR in the full buffer traffic but not necessarily in the bursty traffic.

3) *Round Robin Fashion Scheduling (RRF)*: A user scheduling allocates all RBGs to each one user in turns regardless of their status. New users are appended to the end of the queue.

### B. Workflow of QMIX

For each agent network, it accepts local observation  $o_t^a$  and last local action  $u_{t-1}^a$  as input, then outputs the individual action-value function  $Q_a(\tau^a, u_t^a)$ . A *Gated Recurrent Unit (GRU)* is integrated in the network, which is used for building recurrent neural network (RNN) [?]. RNN is always applied to the sequence processing, such as machine translation and speech recognition [?]. The GRU accepts input  $x_t^a$  and former hidden state  $h_{t-1}^a$ , then generates the output  $y_t^a$  and new hidden

state  $h_t^a$ . The hidden state records the historical information, which is regarded as decision reference. GRU allows the network to not only remember the past information, but also selectively forget some unimportant information. As the GRU has memory function, thus QMIX leverages the hidden state to reflect the action-observation history  $\tau^a$  considering the Dec-POMDP. After the  $Q_a(\tau^a, u_t^a)$  is generated, it is used to make decision follows the  $\epsilon$ -greedy policy:

$$\pi^a(u^a|o^a) = \begin{cases} 1 - \epsilon + \frac{\epsilon}{|\mathcal{U}|}, u^a = \arg \max_{u^a} Q_a(\tau^a, \cdot) \\ \frac{\epsilon}{|\mathcal{U}|}, u^a \neq \arg \max_{u^a} Q_a(\tau^a, \cdot) \end{cases} \quad (30)$$

This policy implies that the agent will randomly select an action with a probability of  $\epsilon$ , or follow the action with maximum action-value with a probability of  $1 - \epsilon$ . Such policy allows agent to make further exploration, because it increases the probability of covering all the possible state-action pairs when apply training. For RBG allocation, the formulation above decides which user should be allocated the RBG controlled by agent  $a$ .

So far it is feasible to produce individual action-value functions for each agent. The next step is to combine them into a centralized action-value function  $Q_{tot}(\tau, \mathbf{u})$ . To ensure the monotonicity constraint in Eq. 24, a mixing network with special weights and biases is proposed to address the problem. As shown in Fig. 5, the mixing network is a fully connected network with two layers. Especially, the weights and biases is generated by the hyper-network at each time step rather than the inherent parameters. The hyper-network accept global state  $s_t$  as input and calculating weights and biases by four branches, where each red block is a linear layer. For instance, if the layer 1 take a weights matrix with shape  $(16, 32)$ , then the hyper-network will generate a vector with  $16 \times 32 = 512$  elements. And the vector will be reshaped into a new matrix which satisfies the requirement of layer 1. Moreover, the branch 1 and branch 3 take absolute activation function to ensure the mixing network weights are non-negative. Also the biases are generated with same method, but it is not restricted to be non-negative. The final bias is generated by a 2 layer neural network with a ReLU activation function. Here gives the proof of Eq. 24 under these settings:

**Theorem 2.**  $\frac{\partial Q_{tot}}{\partial Q_a} \geq 0, \forall a \in \mathcal{A}$ , when the weights of mixing network is non-negative.

*Proof.* Considering the mixing network in Fig. 5, let  $W_1, W_2, B_1, B_2$  denote the generated weights and biases from the hypernetworks,  $\mathbf{Q} = \{Q_1, \dots, Q_n\}$  being the input vector and  $g(x) = \begin{cases} \alpha(e^x - 1), x < 0, \alpha > 0 \\ x, x \geq 0 \end{cases}$  being the ELU activation function. Thus the  $Q_{tot}$  can be written as:

$$\begin{aligned} Q_{tot} &= W_2 g(W_1 \mathbf{Q} + B_1) + B_2 \\ &= W_2 g(Y_1) + B_2 \end{aligned} \quad (31)$$

Then,

$$\begin{aligned} \frac{\partial Q_{tot}}{\partial \mathbf{Q}} &= \frac{\partial Q_{tot}}{\partial g(Y_1)} \cdot \frac{\partial g(Y_1)}{\partial Y_1} \cdot \frac{\partial Y_1}{\partial \mathbf{Q}} \\ &= W_2 \cdot g'(W_1 \mathbf{Q} + B_1) \cdot W_1 \end{aligned} \quad (32)$$

Since

$$g'(x) = \begin{cases} \alpha e^x, & x < 0, \alpha > 0 \\ 1, & x \geq 0 \end{cases} \quad (33)$$

And  $W_1, W_2$  is non-negative, thus:

$$\frac{\partial Q_{tot}}{\partial Q_a} \geq 0, \forall a. \quad (34)$$

□

Through the hyper-network, the global state  $s_t$  is integrated to estimate the centralized action-value function. And the monotonic monotonicity constraint is fully satisfied. Then QMIX can be trained end-to-end by minimizing the following loss:

$$\mathcal{L}(\theta) = [r + \gamma \max_{\mathbf{u}'} Q_{tot}(\tau', \mathbf{u}', s'; \theta) - Q_{tot}(\tau, \mathbf{u}, s; \theta)]^2 \quad (35)$$

where the  $\theta$  is the parameters of deep neural network. Eq. 35 is similar to Eq. 22, along with update operations.

## REFERENCES

- [1] W. Saad, M. Bennis, and M. Chen, "A vision of 6g wireless systems: Applications, trends, technologies, and open research problems," *IEEE Network*, vol. 34, no. 3, pp. 134–142, 2020.
- [2] K. B. Letaief, W. Chen, Y. Shi, J. Zhang, and Y. A. Zhang, "The roadmap to 6g: Ai empowered wireless networks," *IEEE Communications Magazine*, vol. 57, no. 8, pp. 84–90, 2019.
- [3] S. Dang, O. Amin, B. Shihada, and M.-S. Alouini, "From a human-centric perspective: What might 6g be?," *arXiv preprint arXiv:1906.00741*, 2019.
- [4] D. Tse and P. Viswanath, *Fundamentals of wireless communication*. Cambridge university press, 2005.
- [5] X. Liu, E. K. P. Chong, and N. B. Shroff, "Opportunistic transmission scheduling with resource-sharing constraints in wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 19, no. 10, pp. 2053–2064, 2001.
- [6] E. L. Hahne, "Round-robin scheduling for max-min fairness in data networks," *IEEE Journal on Selected Areas in communications*, vol. 9, no. 7, pp. 1024–1039, 1991.
- [7] D. Tse, "Multiuser diversity in wireless networks," in *Wireless Communications Seminar, Stanford University*, 2001.
- [8] A. B. Sediq, R. H. Gohary, R. Schoenen, and H. Yanikomeroglu, "Optimal tradeoff between sum-rate efficiency and jain's fairness index in resource allocation," *IEEE Transactions on Wireless Communications*, vol. 12, no. 7, pp. 3496–3509, 2013.
- [9] G. Zhu, D. Liu, Y. Du, C. You, J. Zhang, and K. Huang, "Toward an intelligent edge: wireless communication meets machine learning," *IEEE Communications Magazine*, vol. 58, no. 1, pp. 19–25, 2020.
- [10] Q. Cao, S. Zeng, M.-O. Pun, and Y. Chen, "Network-level system performance prediction using deep neural networks with cross-layer information," in *ICC 2020-2020 IEEE International Conference on Communications (ICC)*, pp. 1–6, IEEE, 2020.
- [11] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.
- [12] I.-S. Comşa, 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, p. 315, 2019.
- [13] I.-S. Comşa, 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.
- [14] C. Xu, J. Wang, T. Yu, C. Kong, Y. Huangfu, R. Li, Y. Ge, and J. Wang, "Buffer-aware wireless scheduling based on deep reinforcement learning," in *2020 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–6, IEEE, 2020.
- [15] Y. Hua, R. Li, Z. Zhao, X. Chen, and H. Zhang, "Gan-powered deep distributional reinforcement learning for resource management in network slicing," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 2, pp. 334–349, 2019.
- [16] L. Buşoniu, R. Babuška, and B. De Schutter, "Multi-agent reinforcement learning: An overview," in *Innovations in multi-agent systems and applications-1*, pp. 183–221, Springer, 2010.
- [17] R. B. Myerson, *Game theory*. Harvard university press, 2013.
- [18] O. Vinyals, I. Babuschkin, W. M. Czarnecki, M. Mathieu, A. Dudzik, J. Chung, D. H. Choi, R. Powell, T. Ewalds, P. Georgiev, et al., "Grand-master level in starcraft ii using multi-agent reinforcement learning," *Nature*, vol. 575, no. 7782, pp. 350–354, 2019.
- [19] Shapley and S. L., "Stochastic games," *Proceedings of the National Academy of Sciences*, vol. 39, no. 10, pp. 1095–1100, 1953.
- [20] G. E. Monahan, "State of the art—a survey of partially observable markov decision processes: theory, models, and algorithms," *Management science*, vol. 28, no. 1, pp. 1–16, 1982.
- [21] J. D. Williams and S. Young, "Partially observable markov decision processes for spoken dialog systems," *Computer Speech & Language*, vol. 21, no. 2, pp. 393–422, 2007.
- [22] I. Goodfellow, Y. Bengio, A. Courville, and Y. Bengio, *Deep learning*, vol. 1. MIT press Cambridge, 2016.
- [23] D. Michie, D. J. Spiegelhalter, C. Taylor, et al., "Machine learning," *Neural and Statistical Classification*, vol. 13, no. 1994, pp. 1–298, 1994.
- [24] J. Y. Halpern and N. Rong, "Cooperative equilibrium," in *Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: volume 1-Volume 1*, pp. 1465–1466, 2010.
- [25] C. J. Watkins and P. Dayan, "Q-learning," *Machine learning*, vol. 8, no. 3–4, pp. 279–292, 1992.
- [26] K. Hornik, M. Stinchcombe, H. White, et al., "Multilayer feedforward networks are universal approximators," *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989.
- [27] P. Sunehag, G. Lever, A. Gruslly, W. M. Czarnecki, V. F. Zambaldi, M. Jaderberg, M. Lanctot, N. Sonnerat, J. Z. Leibo, K. Tuyls, et al., "Value-decomposition networks for cooperative multi-agent learning based on team reward," in *AAMAS*, pp. 2085–2087, 2018.
- [28] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: Encoder-decoder approaches," *arXiv preprint arXiv:1409.1259*, 2014.
- [29] T. Rashid, M. Samvelyan, C. S. De Witt, G. Farquhar, J. Foerster, and S. Whiteson, "Qmix: Monotonic value function factorisation for deep multi-agent reinforcement learning," *arXiv preprint arXiv:1803.11485*, 2018.