User-Centric Resource Allocation in FD-RAN: A Stepwise Reinforcement Learning Approach

Jiacheng Chen, *Member, IEEE*, Jingbo Liu, and Haibo Zhou, *Senior Member, IEEE*

***Abstract*—To improve resource utilization flexibility and enhance resource cooperation, a novel fully decoupled radio access network (FD-RAN) architecture was conceived, allowing separate resource allocation of uplink and downlink. One of the envisions of FD-RAN and future 6G is to provide personalized services to users, namely, satisfying users' demands differently. To achieve this goal, we utilize the idea from user-centric resource allocation (UCRA), which specifically takes into account users' subjective values of services during resource allocation. We first define a novel user utility function based on the prospect theory. Then, we study a subchannel allocation problem with an underlying heterogeneous network. Confronted with the complex solution space, we develop a stepwise reinforcement learning (RL) method which takes an action for only one user at each step. Furthermore, an action filter is utilized to select only feasible actions that meet the problem's constraints, such that the generated training data samples for RL are all valid, making training more efficient and stable. The method is also extended to multiagent case, where users can choose their actions with their own agents. Owing to the stepwise action process, the nonstationary environment problem in standard multiagent RL is naturally avoided. As a result, our method can be scaled to more agents. We have performed extensive simulations and the results validate the effectiveness of our proposed methods.**

***Index Terms*—Deep reinforcement learning (DRL), fully decoupled radio access network (FD-RAN), multiagent reinforcement learning (RL), UL/DL decoupling, user-centric resource allocation (UCRA), value of services.**

I. Introduction

FLEXIBILITY of resource allocation is critical for enabling cooperation among multiple heterogeneous base stations (BSs), such that both resource utilization efficiency and users' service quality can be improved. For this purpose, the fully decoupled radio access network (FD-RAN) [1] was developed by Yu et al. as a novel 6G paradigm which considers uplink and downlink separately, promising more flexible association and resource allocation for each link [2].

The benefits of FD-RAN need to be embodied through a well-designed resource allocation method, which essentially addresses the conflict between users' various demands for services and the network's supply of transmission resources. Traditionally, resource allocation is conducted from the network's perspective, either optimizing the conventional performance indicators, such as throughput, latency, Quality of Service (QoS), and coverage rate [3] for the network, or maximizing system utility defined as function of revenues [4]. By involving the concepts like Quality of Experience (QoE) and user utility, users are further taken into account in resource allocation. However, the definitions of these concepts are still based on the conventional performance indicators for users [5], [6].

Given that 5G has focused on vertical industry applications, we think 6G should provide personalized service for each individual user [7], [8]. As a matter of fact, when users request services through the network, what they actually obtain are the values brought by these services. These values are subjective to users, thus they can be different for different users even if they request the same services. Therefore, users' subjective values of requested services should be considered in resource allocation. Appropriately allocating resources to achieve more total values is a common principle for economists. It can also be explained through a special case: if one user is requesting a remote surgery service while another is requesting a gaming service, then the first user should have a higher priority in resource allocation, since remote surgery is more valuable to the first user than gaming to the second user. Unfortunately, in the current network, due to the oversight of users' values of services in resource allocation, the total values delivered from consuming the network resources are not optimized, and users cannot experience personalized services even if they value their services differently.

To this end, user-centric resource allocation (UCRA) [7] has been proposed, which takes into account users' subjective values of services, and can be optimized to deliver more total values to users. UCRA enables personalized service transmission through investigating users' true demands, i.e., the extent of desire for the requested services. Yet, it is still very challenging in several aspects to realize UCRA in a network like FD-RAN, considering the flexibility of resource allocation and cooperation among multiple heterogeneous BSs.

Manuscript received 5 January 2024; revised 24 February 2024 and 31 March 2024; accepted 12 April 2024. Date of publication 16 April 2024; date of current version 26 June 2024. This work was supported in part by the National Natural Science Foundation Original Exploration Project of China under Grant 62250004; in part by the Natural Science Foundation of China (NSFC) under Grant 62001259 and Grant 62271244; in part by the Innovation and Entrepreneurship of Jiangsu Province High-Level Talent Program; in part by the Summit of the Six Top Talents Program of Jiangsu Province; in part by the Major Key Project of PCL; and in part by the Basic and Frontier Research Project of PCL. *(Corresponding author: Haibo Zhou.)*

Jiacheng Chen is with the Department of Strategic and Advanced Interdisciplinary Research, Peng Cheng Laboratory, Shenzhen 518000, China (e-mail: chenjch02@pcl.ac.cn).

Jingbo Liu is with the School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: liujingbo@sjtu.edu.cn).

Haibo Zhou is with the School of Electronic Science and Engineering, Nanjing University, Nanjing 210023, China (e-mail: haibozhou@nju.edu.cn).

Digital Object Identifier 10.1109/JIOT.2024.3389208

2327-4662 © 2024 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: Nanjing University. Downloaded on April 09,2025 at 11:23:54 UTC from IEEE Xplore. Restrictions apply.

First of all, we need a way to model the user's extent of desire for services. This is accomplished through defining a novel user utility function, which characterizes a user's subjective view on a specific service through a profile containing two factors, namely, the minimal data rate requirement and the value of service. Further, our approach treats the uplink and downlink of a service separately since they are often not symmetrical with respect to these two factors. For instance, during an online video conference, uplink is more important for the presenter while downlink is more important for the listener. Adopting the decoupling of uplink and downlink is also for exploiting the flexibility of resource allocation in FD-RAN. With the above considered variables, we then need a mathematic function for passing in these variables to calculate the utility or disutility that a user will obtain after resources are allocated. We resort to prospect theory [9] in the field of behavior economics. Intuitively, prospect theory summarizes the properties of utility functions that align with most human behaviors, i.e., diminishing gain, diminishing loss and loss aversion.

Given the user utility function, we study a subchannel allocation problem so as to optimize the sum utility of all users. We consider an underlying heterogeneous network (HetNet) consisting of macro BSs (MBSs) with larger coverage and small BSs (SBSs) with higher density. The flexibility of resource allocation is further embodied in that the uplink and downlink of a user can be associated with any BS. Such kind of HetNets can optimize the resource utilization efficiency through more flexible link association [10], [11].

However, flexibility brings a larger and more complicated solution space, meaning that it becomes more difficult to optimize subchannel allocation. Besides, although we propose a specific user utility function in this article, we still want our approach for resource allocation to be generalized to other potential forms of user utility definitions. For such sakes, we employ deep reinforcement learning (DRL), which has been widely used to solve various resource allocation problems [12], [13], [14], [15]. Rather than developing an algorithm for each specific problem, DRL treats the problem as a black box environment and leverages reinforcement learning (RL) to train a deep neural network through interactions with the environment. In this way, our DRL-based approach can be easily adapted for other subchannel allocation problems with various user utility definitions.

The main challenges for adopting DRL-based approaches lie in how to properly embed the original problem into the DRL framework. This requires providing problem-specific definitions of state, action, reward, episode, and agent. Furthermore, since our subchannel allocation problem is constrained, impossible actions that violate constraints must be carefully handled. Additionally, the action space should be reduced to a much lower level than the space of decision variables so as to simplify and stabilize DRL training.

To overcome these difficulties, we develop a stepwise RL method where only one user's decision variable is determined per step in an episode. This technique greatly reduces the action space. Further, the action space is no longer correlated with the number of users, allowing our method to be scaled to more users. To make the stepwise RL feasible, we encode the actions taken by all users (i.e., the global state of actions) in the state vector. We apply an action filter at each step so that only feasible actions can be selected. Then, all generated training data samples from the environment are valid and useful, so training efficiency and stability can be enhanced. Building on this single-agent version, we also develop the multiagent stepwise RL where users' actions are determined by different agents, namely, by different neural networks with their own parameters (weights). Owing to the stepwise learning technique, the nonstationary environment problem in standard multiagent RL can be avoided naturally, as there will be no joint actions determining the next environment state together. As an overview, the main contributions of our study are summarized as follows.

1) To provide personalized data transmission services to users in FD-RAN, we investigate UCRA, and propose a novel definition of user utility that specifically takes into account users' different subjective values on requested services.
2) We then formulate a subchannel allocation problem that optimizes the sum utility of all users. The problem is studied under a HetNet, while considering flexible link association and resource allocation.
3) To solve the problem, we develop a stepwise RL method to reduce the dimension of action space. To facilitate the stepwise process, we include the global state of actions in state vector. We also utilize action filter as a critical technique to meet the constraints of the problem and guarantee the usability of training data samples.
4) We further extend the stepwise RL to multiagent case. Compared with standard multiagent RL, the nonstationary environment problem is avoided naturally, so the proposed method can be scaled to more users.

The remainder of this article is organized as follows. In Section II, we present a review of relevant literature on utility-based resource allocation, and DRL based resource allocation. Section III introduces the fully decoupled network model and problem formulation. The stepwise RL method and its extension to multiagent case are elaborated in Section IV, which is followed by Section V showcasing the simulation results and analyses. Finally, we draw the conclusions in Section VI.

II. Related Works

A. Utility-Based Resource Allocation

There have been a wide range of objectives for wireless resource allocation, e.g., system throughput, fairness, energy efficiency, QoS/QoE, profit, just to name a few. In some literatures, a system/network utility is also defined. Lin et al. [3] studied the user association and spectrum allocation problem in HetNet. A proportionally fair utility function based on the coverage rate is defined, and is used as the objective of optimization. In [5], a QoS-aware joint user association, resource and power allocation approach is proposed. The objective is to maximize the number of serving users, under the constraint that users' QoS demands are satisfied. The metric of

Authorized licensed use limited to: Nanjing University. Downloaded on April 09,2025 at 11:23:54 UTC from IEEE Xplore. Restrictions apply.

QoS is chosen as the minimum required data rate for reliable communications to users. Luong et al. [4] summarized the objectives for applying economic and pricing models in 5G, including system utility. Here, system utility is described as the sum utility of both users and network operators. The former is defined based on data rate, and the latter is based on revenue. Hence, the motivation is to consider the benefit of both sides when designing resource allocation schemes. In another work, Tadayon and Aissa [6] designed an auction-based radio resource allocation. In the theory of auction, users bid with a valuation of the item. Tadayon and Aissa [6] mentioned that it is natural to use instantaneous spectral usage as the valuation. Tan et al. [16] explained the utility from economics and defines utility as a mapping from allocated bandwidth to users' satisfaction level. This article further formulate utility according to specific traffic type, such as multimedia, and give the shape of utility for hard QoS, soft QoS, and best effort QoS traffic. In [17], a post-disaster resource allocation is studied, and utility is used to reflect the urgency of requirement, i.e., exigency. Different from these works, we explicitly consider users' subjective (personalized) values of the requested services in the utility. Thus, we introduce a new user-specific dimension to the metrics considered in resource allocation. The advantages of the utility definition in this article are threefold. First, when doing resource allocation, the total value that the network brings to users can be optimized, and the network can understand user's true demands so as to provide personalized services to different users. Second, uplink and downlink are considered differently to accommodate the services whose uplink and downlink are asymmetric, as well as to fit the decoupling of uplink and downlink in the fully decoupled HetNet. Last, prospect theory is utilized to shape the utility function, such that it aligns with most human behaviors, i.e., diminishing gain, diminishing loss and loss aversion.

B. Deep Reinforcement Learning-Based Resource Allocation

With the success of a variety of deep learning methods, a lot of research works have attempted to enhance the wireless cellular network with intelligence [18], [19]. Among all such efforts, DRL has been one of the most popular approaches for solving various kinds of problems in wireless communications, including resource allocation. Chang et al. [20] introduced a distributive dynamic spectrum access approach. The secondary users adopts DRL to learn spectrum access strategies based on the current and past spectrum sensing outcomes. recurrent neural network (RNN) is utilized to capture the underlying temporal correlation. Another work [14] studies a general multichannel access problem and derives distributed access actions via multiuser DRL so as to maximize the network utility. He et al. [12] proposed a channel assignment problem in the nonorthogonal multiple access (NOMA) system, where each channel can be multiplexed by two users with different power allocation. An attention-based neural network is used with RL to perform channel assignment.

DRL has also been applied for resource allocation in HetNets [21], [22], device-to-device (D2D) networks [23], [24], and vehicular/vehicle-to-vehicle (V2V) networks [25], [26], [27]. Specifically, Liang et al. [26] used a fingerprint-based deep Q-network (DQN) for distributed resource management in high-mobility vehicular environments. Each V2V link is treated as an agent, and learns to find spectrum and power allocation. In order to stabilize the multiagent training, the authors utilize a technique in [28] by adding fingerprints (training iteration number and rate of exploration) in the state vector. However, only four V2V links are considered in the simulation, so it is unknown whether the multiagent DRL (MADRL) can be scaled to more users. To tackle the nonstationary issue in multiagent environment, Xiang et al. [27] utilized several techniques, including hysteretic Q-learning and concurrent experience replay trajectory (CERT).

To summarize, the following design choices have to be made when adopting DRL: 1) the neural network architecture, such as fully connected, convolutional neural network (CNN), RNN, and graph neural network (GNN); 2) the RL architecture, such as DQN and actor–critic (AC); and 3) the decomposition of action space, such as single agent and multiagent. Different combination of the above choices will lead to a variety of DRL-based methods. The novelty of our stepwise RL belongs to the third choice. Specifically, in single-agent case, the dimension of action space is reduced to constant by taking action for users in a stepwise way and encoding the global state of actions in the state vector. Also, a filter can be applied during the process of taking actions so as to keep feasible actions only. Further, in multiagent case, the nonstationary environment problem is naturally avoided, thus the method can be scaled to more users. Note that in this article we mainly compare with the standard multiagent RL given in the third choice. Novel algorithms belonging to the first and second choices can be easily integrated with our proposed methods.

III. System Model and Problem Formulation

A. System Model

We consider a fully decoupled HetNet with M_{MBS} macro base stations and M_{SBS} small base stations serving N users in total. We also assume $M_{\mathrm{MBS}} < M_{\mathrm{SBS}}$, indicating that the density of SBSs is higher than MBSs. Thus, users are more likely to be closer to SBSs, while they are within the coverage of MBSs. The network allows decoupled access of uplink and downlink for users. Put it another way, uplink and downlink traffic of the same user can be served (but not necessarily) by different BSs. We can easily derive the four possible cases for a user to access BSs for both uplink and downlink transmission, as illustrated in Fig. 1. Decoupled access is a natural mode in the FD-RAN proposed by Yu et al. [1], where BSs are physically decoupled into three types, namely, control BSs, uplink BSs, and downlink BSs. Uplink and downlink can be treated as separate links, leading to flexibility. Thus, the heterogeneous BSs and network resources can be utilized more efficiently. Our network model is general in terms of user association pattern, so it is straightforward to apply our proposed methods to other conventional network architectures. We assume that MBSs (SBSs) have K^u_{MBS} and K^d_{MBS} (K^u_{SBS}

Authorized licensed use limited to: Nanjing University. Downloaded on April 09,2025 at 11:23:54 UTC from IEEE Xplore. Restrictions apply.

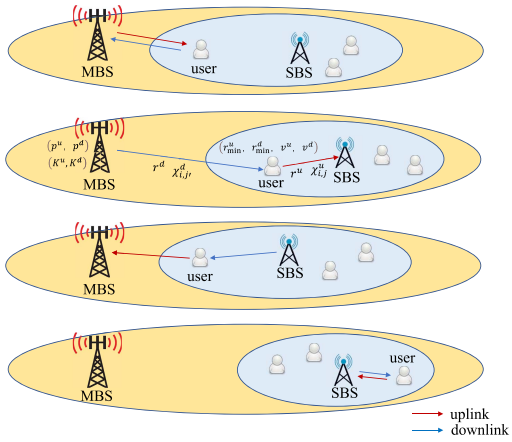


Fig. 1. Illustration for the four possible cases of decoupled access.

and K^d_{SBS}) orthogonal subchannels for uplink and downlink transmission, respectively.

In this article, we focus on joint BS selection and subchannel allocation, which belongs to the MAC layer function. Generally, the MAC layer takes the physical layer measurement as input, and outputs the corresponding resource allocation results. We mainly consider the signal-to-noise ratio (SNR) from physical layer, an important indicator typically used for determining the serving BS and allocated subchannels of a user. In this way, the MAC layer does not have to know the details of the physical layer, e.g., how the signal propagates through the channel with various fading effects, since all the details are finally represented through the SNR. Then, the physical layer model can be represented through the Shannon formula, which gives the transmission rate of a link as

$$r = \mathrm{BW}\log_2\left(1 + \frac{gP_T}{P_N}\right) \tag{1}$$

where BW is the total bandwidth of allocated subchannels for the link, P_T is the transmit power, and P_N is the noise power, and g denotes the channel gain. Without loss of generality, we assume that each user has only one serving BS for uplink and one serving BS for downlink.

B. Problem Formulation

We first model the utility of users when using the network. In the following, if not explicitly mentioned, the defined variables are used to describe a certain user. Also, since access is decoupled, we consider uplink and downlink separately and use different superscripts. In fact, this is more realistic because uplink and downlink are not symmetric in most cases. We only consider duplex services, which require both uplink and downlink data rates.

The transmission rate demands of users are modeled by minimum rate demands $r^u_{\min}$ and $r^d_{\min}$ for uplink and downlink, respectively. The minimum rate demands model users' basic usage of the network. However, the actual rates r^u and r^d allocated to users can be much higher, and users can experience far beyond their basic usage. For example, users can watch videos with low definition (e.g., 480p), but they can also watch with a higher definition (e.g., 1080p) if the available rates are larger. The basic usage can also be regarded as a bundle of indispensable apps, such as mail, instant messaging, and news recommendation, but more apps such as social media and video streaming can be supported by more allocated rates.

Users use the network to request various kinds of services, because these services can bring value to them. Hence, in UCRA [7], resources are allocated in an economically efficient manner, that is, to the services that can bring more value to users. From the perspective of network, it provides personalized services to users. We use v^u and v^d to denote the values of users' requested services for uplink and downlink, respectively. The value is essentially the maximum price that users are willing to pay for acquiring the services. Correspondingly, the actual prices for using the BSs at the current time are denoted by p^u and p^d for uplink and downlink, respectively. By mentioning the current time, we mean that the prices can be dynamic. Dynamic pricing of mobile data has been studied, such as [29] and our previous work [30].

Given the above definitions, we use the 4-tuple profile $(r^u_{\min}, v^u, r^d_{\min}, v^d)$ to characterize the requested services of users. Then, the user utility is given by

$$\begin{aligned}&\mathrm{Utility}\left(r^u, r^d, p^u, p^d\right)\\ &\quad= \omega f\left(S_1(r^u - r^u_{\min})\right) + (1-\omega) f\left(S_1(r^d - r^d_{\min})\right)\\ &\qquad+ \omega f\left(S_2(v^u - p^u)\right) + (1-\omega) f\left(S_2(v^d - p^d)\right)\end{aligned} \tag{2}$$

where $f(\cdot)$ is the utility function, $S_1(\cdot), S_2(\cdot)$ are the scaling functions that confine the output inside a specific region, and $\omega \in [0, 1]$ is a variable that weights the relative importance between uplink and downlink. Furthermore, the utility function $f(\cdot)$ is given by prospect theory [9], and have the properties of diminishing gain, diminishing loss and loss aversion. A classic utility function is given below

$$f(x) = \begin{cases} x^\alpha, & x \geq 0 \\ -\lambda(-x)^\beta, & x < 0 \end{cases} \tag{3}$$

where α, β, λ are parameters that shape the curve of utility function. $\alpha, \beta \in (0, 1)$ reflect the diminishing gain (when $x > 0$) and diminishing loss (when $x < 0$) property, respectively. A smaller value means that the gain/loss diminishes faster. $\lambda > 1$ reflects the loss aversion property, and a larger value indicates more significant loss compared with gain.

The user utility defined above takes into account two major factors, namely, transmission rate and value of services. Specifically, if the actual rate is higher than the minimum rate demand, then users' utility will increase, but the speed of increasing will become slower as the rate gets larger. Otherwise, if the actual rate is lower than the minimum rate demand, then the utility becomes negative and decreases sharply at the beginning. Similarly, if the value of requested services is larger than the current price, then users gain utility, otherwise users gain disutility. The user's profile will also influence the resources allocated to the user. From (2), a larger value of service will lead to more user utility, thus more resources will be allocated to the user. Also, a lower minimum rate demand means less resources are required to gain positive utility, so resources will be allocated with higher priority. Users can set the minimum rate demand and value for various requested services through specifically designed

Authorized licensed use limited to: Nanjing University. Downloaded on April 09,2025 at 11:23:54 UTC from IEEE Xplore. Restrictions apply.

interfaces such as cybertwin [7], [31]. Users can also directly feedback their experience to the network in simple ways. For example, if the user feels not satisfactory, then a high-level feedback is delivered to the network. Then, the value of service parameter in the user's utility will be increased, such that more resources can be potentially allocated to the user.

We study a general subchannel allocation problem in the decoupled-access HetNet scenario described above. The objective is to maximize the sum utility of all users. With such an objective of resource allocation, not only the transmission efficiency is considered, but also the total value delivered to users is taken into account. Especially, the latter is not considered in traditional networks that does not use the UCRA criterion. We use $\mathcal{N} = \{1, \ldots, N\}$ to denote the set of all users, and $\mathcal{M} = \{1, \ldots, M\}$ to denote the set of all BSs. A user is represented by $i, i \in \mathcal{N}$, and the BSs that serve the uplink and downlink of the user are represented by $j \in \mathcal{M}$ and $j' \in \mathcal{M}$, respectively. $\mathcal{N}_j^u$ and $\mathcal{N}_{j'}^d$ represent the set of users whose uplink and downlink is served by BSs j and j', respectively. Furthermore, we use $\chi_{i,j}^u$ and $\chi_{i,j'}^d$ as decision variables to represent the number of uplink and downlink subchannels allocated to user i. Users are either allocated both uplink and downlink subchannels or nothing. If a user is not allocated uplink or downlink subchannels, then we set j or j' to 0, and use $\chi_{i,0}^u$ and $\chi_{i,0}^d$ instead. For users who are not allocated subchannels (i.e., their actual rates are zero), we only calculate the transmission rate part of utility, since they do not pay any price for the services that are not transmitted. The problem formulation is given below.

Objective:

$$\max \quad \sum_{i \in \mathcal{N}} \text{Utility}_i\left(r_i^u, r_i^d, p_j^u, p_{j'}{}^d\right). \tag{4}$$

s.t.

$$\sum_{i \in \mathcal{N}_j^u} \chi_{i,j}^u \leq K_j{}^u \quad \forall j \in \mathcal{M} \tag{5}$$

$$\sum_{i \in \mathcal{N}_{j'}^d} \chi_{i,j'}^d \leq K_{j'}{}^d \quad \forall j' \in \mathcal{M} \tag{6}$$

$$\left(\chi_{i,j}^u > 0 \wedge \chi_{i,j'}^d > 0\right) \vee \left(\chi_{i,0}^u = 0 \wedge \chi_{i,0}^d = 0\right) \quad \forall i \in \mathcal{N}. \tag{7}$$

Constraints (5) and (6) mean that the total number of allocated uplink and downlink subchannels cannot exceed the total subchannels of the BSs. Constraint (7) indicates that a user is either allocated both uplink and downlink subchannels or no subchannels at all.

IV. Stepwise Reinforcement Learning and Multiagent Extension

The formulated problem given by (4)–(7) is a classical combinatorial optimization problem. In practice, such a kind of problem is usually solved by heuristic algorithms such as greedy algorithms. The time complexity depends on the number of iterations conducted in these algorithms. Also, the results are mostly suboptimal. Furthermore, the designed algorithms are often specific to the formulated certain problem, and cannot be generalized to other similar resource allocation problems. To this end, in this article, we resort to the DRL [32] framework and attempt to present a general solution for such a kind of combinatorial optimization problems. Another major motivation for using DRL is that the utility given by (2) as well as the optimization objective given by (4) can be changed with another criteria of resource allocation. In what follows, we first highlight the key points of DRL and introduce the DQN family, leading to the expressions of D3QN, which will be used by our stepwise RL in the following. Then, we present our proposed stepwise RL methods for solving the subchannel allocation problem, consisting of a single-agent version and a multiagent extension.

A. Deep Reinforcement Learning and DQN Family

RL [33] solves a kind of problems that can be modeled by the Markov decision process (MDP). The mathematical definition MDP has been extensively introduced in literature, so we only explain its structure. In MDP, an agent takes an action in the environment at each step, such that the environment changes due to the action, and the agent obtains a feedback (a.k.a reward) from the new environment. Specifically, the 4-tuple $\xi = (\mathbf{s}, a, \mathbf{s}', r')$ can be used to describe the above process, where $\mathbf{s} \in \mathcal{S}$ is the state vector representation of the current environment and $\mathcal{S}$ denotes the state space containing all possible values of $\mathbf{s}$, $a \in \mathcal{A}$ is the action that is taken only based on $\mathbf{s}$ and $\mathcal{A}$ denotes the action space, $\mathbf{s}'$ is the next state given $\mathbf{s}$ and a, and $r' \in \mathbb{R}$ is the reward derived from the new state $\mathbf{s}'$. The goal of RL is to find an optimal policy π:$\mathbf{s} \rightarrow a$ for the agent that maximizes the sum of discounted rewards R obtained from all steps in an episode. R is given by

$$R = \sum_{l=0,1,2,\cdots,L_{ep}} \gamma^l r_{l|\pi}{}' \tag{8}$$

where l is the step index, L_{ep} denotes the last step of an episode, $r_{l|\pi}{}'$ is the reward obtained from step l after taking action by following a policy π, and γ is the discounting factor.

Traditional RL algorithms, such as Q-learning, utilize a Q-table to store the Q-value $Q(\mathbf{s}, a)$ of each state–action pair and updates the Q-table by the Bellman equation. However, the size of Q-table will become too large when the state space increases, such that these algorithms are no longer efficient. To deal with the problem, deep learning has been integrated with RL due to its advantage in generalization. The basic idea is to use a deep neural network to replace the Q-table and model the Q-value function directly. The input of the neural network is $\mathbf{s}$, and outputs are the Q-values of all actions. The neural network is also trained based on the Bellman equation.

The most famous DRL architecture is DQN [34]. DQN exploits two important techniques. One is experience replay, which saves a large number of training samples with a memory buffer. Then, training is conducted by randomly selecting a mini-batch of samples from the memory buffer and performing parameters update with stochastic gradient descent. Another is introduction of the target network, which updates its parameters for every τ steps by copying from the evaluation network. The loss function for DQN is given by

$$\text{loss}(\boldsymbol{\theta}) = \mathbb{E}\left[\left(r' + \gamma \max_{a'} Q(\mathbf{s}', a'|\boldsymbol{\theta}') - Q(\mathbf{s}, a|\boldsymbol{\theta})\right)^2\right] \tag{9}$$

Authorized licensed use limited to: Nanjing University. Downloaded on April 09,2025 at 11:23:54 UTC from IEEE Xplore. Restrictions apply.

TABLE I
6-BIT ENCODING OF ACTION STATE FOR A USER

bit value	1st bit	2nd bit	3rd bit	4th bit	5th bit	6th bit
0	has not taken action	allocated subchannels	uplink: SBS	downlink: SBS	uplink subchannels: 1	downlink subchannels: 1
1	has taken action	allocated no subchannels	uplink: MBS	downlink: MBS	uplink subchannels: 2	downlink subchannels: 2

where $\boldsymbol{\theta}$ and $\boldsymbol{\theta}'$ are the parameters of evaluation network and target network, respectively. Further, the double DQN (DDQN) [35] improves DQN by selecting the best action of state $\mathbf{s}'$ using the evaluation network (parameterized by $\boldsymbol{\theta}$) instead of the target network (parameterized by $\boldsymbol{\theta}'$). The loss function of DDQN is given by

$$\text{loss}(\boldsymbol{\theta}) = \mathbb{E}\Big[\big(r' + \gamma Q(\mathbf{s}', \arg\max_{a'} Q(\mathbf{s}', a'|\boldsymbol{\theta})|\boldsymbol{\theta}') - Q(\mathbf{s}, a|\boldsymbol{\theta})\big)^2\Big]. \tag{10}$$

By using the dueling network architecture [36], the original network $Q(\mathbf{s}, a|\boldsymbol{\theta})$ is replaced by two networks, namely, $V(\mathbf{s}|\boldsymbol{\theta}_V)$ for estimating the state value and $A(\mathbf{s}, a|\boldsymbol{\theta}_A)$ for estimating the state–action value. Then, the representation of Q-value becomes

$$Q(\mathbf{s}, a|\boldsymbol{\theta}_V, \boldsymbol{\theta}_A) = V(\mathbf{s}|\boldsymbol{\theta}_V) + \left(A(\mathbf{s}, a|\boldsymbol{\theta}_A) - \frac{1}{|\mathcal{A}|}\sum_{z\in\mathcal{A}} A(\mathbf{s}, z|\boldsymbol{\theta}_A)\right). \tag{11}$$

The combining of (10) and (11) is also known as D3QN, which exploits all the advantages. For clarity, we still use the notation of $Q(\mathbf{s}, a|\boldsymbol{\theta})$, where $\boldsymbol{\theta} = \boldsymbol{\theta}_V \cup \boldsymbol{\theta}_A$.

B. Stepwise Reinforcement Learning

In order to solve the subchannel allocation problem described in Section III-B, we need to know the number of uplink and downlink subchannels allocated to each user. In practical networks, the overheads of connection establishment between BSs and UEs usually limit the potential number of BSs that the UEs can access. For example, in 3GPP standards, only dual-connectivity is supported currently. Thus, for now, we assume that a user is only accessible to one MBS and one SBS. We also limit the number of allocated subchannels for each link to 2, considering the fairness of resource allocation among users. Then, there are total 16 different cases of decision variable, including a special case, in which no subchannel is allocated to the user. In this way, we have reduced the dimension of the problem's action space from $\mathcal{O}(NMK)$ to $\mathcal{O}(N)$, so as to make the training of DRL more practical.

For the single-agent DRL, the agent needs to determine the decision variables for all users with a single D3QN. Although we have reduced the problem's action space, it is still proportional to the number of users N, which means the DRL-based solution cannot scale when N becomes larger. To this end, we develop a stepwise subchannel allocation method. Instead of determining all the decision variables simultaneously, the agent subsequently determines the decision variable for each user with a fixed sequence. Thus, an episode contains N steps, and in the lth step, an action is selected such that user l's associated BSs and number of allocated subchannels for both uplink and downlink are determined. The action space is then reduced to the space of decision variable for a single user, which is a constant size 16 given the above settings on BS association and subchannel allocation. Hence, the DRL-based solution becomes scalable. The reason for fixing the sequence of allocating subchannels to users is that the action taken in the current step can be inherently mapped to the user.

For a specific user, the state vector consists of three parts. The first part contains the 4-tuple $(r^u_{\min}, v^u, r^d_{\min}, v^d)$ that describes the user's service requests and values. The second part is formed by the SNRs with the user's accessible MBS/SBS for uplink and downlink, namely, $\text{SNR}^u_{\text{MBS}}$, $\text{SNR}^d_{\text{MBS}}$, $\text{SNR}^u_{\text{SBS}}$, $\text{SNR}^d_{\text{SBS}}$. In the last part, we include the state of action of the user, which is encoded with 6 bits, as given in Table I. The first bit indicates whether the user's action has been taken or not. The second bit indicates whether the user has been allocated subchannels or not. The third to sixth bits indicate the specific action taken for the user, representing the user's associated BSs and allocated subchannels. For the stepwise RL, keeping the global states of action for all users is necessary, such that the agent can take action with the knowledge of previous actions. The overall state vector is the concatenation of the state vectors of all users. Thus, the size of state vector is $14N$. Note that we consider the prices of BSs are stationary for a relatively long period, so the state vector does not include the prices of BSs.

In an episode, at each step, an action should be taken. During the training process, the selection of action follows the ε-greedy strategy, with ε decaying at a constant rate. However, the action is not selected from the whole action space. Instead, we apply an action filter, which outputs the feasible actions at the current step, i.e., actions satisfying (5) and (6). A conventional strategy for handling the case when an impossible action is selected is to return a large negative reward. However, such a strategy has several drawbacks. First, since the action is impossible, the 4-tuple $\xi = (\mathbf{s}, a, \mathbf{s}', r')$ saved in the memory buffer is a less valuable sample for training D3QN. Second, the episode has to be terminated early as soon as an impossible action is chosen. As a result, the total samples will be much fewer given a certain number of episodes. Third, even after the D3QN is trained, there is still a chance that an impossible action is selected, so the trained D3QN cannot guarantee a feasible output. Last, as a hyperparameter, the value of large negative reward is difficult to set. If the value is too large, then it will be back propagated and make huge influence, such that the agent will become conservative on taking actions that allocate more subchannels,

Authorized licensed use limited to: Nanjing University. Downloaded on April 09,2025 at 11:23:54 UTC from IEEE Xplore. Restrictions apply.

Algorithm 1: Single-Agent Stepwise RL

1 Initialize: $i_{step} \leftarrow 0$, $\mathcal{N}_{seed} \leftarrow$ seeds, $i_{seed} \leftarrow 0$;
2 **for** *episode* $= 1, \ldots, N_{ep}$ **do**
3 Reset environment randomly with seed i_{seed};
4 **for** *user* i *in* $\mathcal{N}$ **do**
5 $\mathbf{s} \leftarrow$ current state;
6 $\mathcal{A}_f \leftarrow$ action filter of user i under $\mathbf{s}$;
7 $a \leftarrow$ selected action from $\mathcal{A}_f$ with ε-greedy;
8 $\mathbf{s}' \leftarrow$ updated state by taking action a on user i;
9 $r' \leftarrow$ reward from (2) with user i;
10 $\mathcal{A}'_f \leftarrow$ action filter of user $i+1$ under $\mathbf{s}'$;
11 Save $(\mathbf{s}, a, \mathbf{s}', r', \mathcal{A}'_f)$ into memory buffer;
12 **if** $i_{step} > N * |\mathcal{N}_{seed}|$ & $i_{step}\%2 = 0$ **then**
13 Retrieve a mini-batch of samples randomly from memory buffer;
14 Calculate $Q(\mathbf{s}, a|\boldsymbol{\theta})$;
15 Calculate $Q(\mathbf{s}', a'|\boldsymbol{\theta})$ for all $a' \in \mathcal{A}$, and set $Q(\mathbf{s}', a'|\boldsymbol{\theta})$ to a large negative value for $a' \in \mathcal{A} - \mathcal{A}'_f$;
16 Calculate $loss(\boldsymbol{\theta})$ by (10);
17 Update $\boldsymbol{\theta}$;
18 Update $\boldsymbol{\theta}' \leftarrow \boldsymbol{\theta}$ for every τ updates of $\boldsymbol{\theta}$;
19 **end**
20 $i_{step} \leftarrow i_{step} + 1$, $i_{seed} \leftarrow (i_{seed} + 1)\%|\mathcal{N}_{seed}|$;
21 **end**
22 **end**

leading to decrease in rewards. On the contrary, the action filter does not have the above drawbacks. Besides, when calculating the loss function given by (10), we explicitly set the Q-values of impossible actions to a sufficiently large negative value, which is a more appropriate value for impossible actions, so that $\arg\max_{a'} Q(\mathbf{s}', a'|\boldsymbol{\theta})$ returns a feasible action. After an action from action filter is chosen for the step, the state vector updates, and a reward is given. The reward is simply the utility of the user corresponding to the action, given by (2), where r^u, r^d are calculated from (1) with BW equals to the bandwidth of allocated subchannels, and p^u, p^d are the prices of the associated MBS/SBS. The complete training process of stepwise RL is given in Algorithm 1.

C. Adopting Standard Multiagent RL

MADRL is used for solving problems that involve multiple agents taking actions simultaneously. We discuss how to adopt the standard MADRL for our subchannel allocation problem. Each agent is responsible for taking action for only one user, thus the action space for each agent can be reduced to constant as single-agent stepwise RL. For an agent, an episode only contains one single step, namely, allocating subchannels for the corresponding user. All the agents take actions simultaneously, and the subchannel allocation for all users are finished. Each agent only needs to use the state vector for a single user defined above, so the state of action part is no longer required. Since the agents are homogeneous, we adopt a technique called parameter sharing, such that all the agents actually use the same set of parameters. Therefore, only a single D3QN is required, and the training costs will be significantly reduced. However, in order to distinguish each agent, an additional part that encodes different agent's ID is added to the state vector. Then, the state vector is comprised of 4-bit encoding of agent ID (for $N \leq 16$), 4-tuple $(r^u_{\min}, v^u, r^d_{\min}, v^d)$, and 4 SNR values $\text{SNR}^u_{\text{MBS}}, \text{SNR}^d_{\text{MBS}}, \text{SNR}^u_{\text{SBS}}, \text{SNR}^d_{\text{SBS}}$.

At the beginning of an episode, joint actions $a_1, \ldots, a_N$ are chosen by agents. Then, the environment is updated by applying these actions, and (5) and (6) are checked, since there is no action filter that guarantees the feasibility of actions. If the constraints are violated, i.e., the total allocated subchannels exceed the total available subchannels, then the reward for all agents is set to an appropriate negative value as a penalty. Otherwise, the reward for all agents is the sum utility of all users. With parameter sharing, each agent's state $\mathbf{s}_i$, action a_i, and reward r' (same for all agents) are saved as training samples in the memory buffer, and are used for updating the shared parameters during the learning steps.

D. Multiagent Stepwise Reinforcement Learning

However, one of the main challenges in MADRL is the nonstationary environment [32]. Specifically, the next state $\mathbf{s}'$ of environment as well as the reward r' to all agents depends on the joint actions $a_1, \ldots, a_N$ taken by all agents. Thus, from the perspective of each agent, the upcoming state and reward can be different even the agent's action is the same. Then, it becomes difficult to train a stable neural network for each agent.

To this end, we still utilize the stepwise subchannel allocation, and adapt it into the multiagent RL framework. Compared with the single-agent counterpart that relies on a single D3QN to generate the actions of all users, each user is represented by an individual agent, and each agent only needs to decide its own action. The parameters of each agent are denoted by $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_N$, respectively. Basically, with more agents, which mean more parameters and more training episodes to converge, the performance of multiagent case will be improved compared with the single-agent case. Furthermore, each agent takes action in turn instead of simultaneously, so that the nonstationary environment problem is avoided. Besides, unlike the single-agent case that uses a fixed user sequence, the sequence can be changed among different episodes in the multiagent case, since each agent is responsible for taking action for only one user.

We still use the same state vector as defined in the single-agent case for each agent. In en episode, at one step, an agent takes an action, then state is updated and reward is feedback to the agent. Thus, a training sample is saved for the agent. The loss function for an agent i is given by

$$\text{loss}(\boldsymbol{\theta}) = \mathbb{E}\Big[\big(r' + \gamma Q(\mathbf{s}', \arg\max_{a'} Q(\mathbf{s}', a'|\boldsymbol{\theta}_{i+1})|\boldsymbol{\theta}'_{i+1}) - Q(\mathbf{s}, a|\boldsymbol{\theta}_i)\big)^2\Big]. \tag{12}$$

From the above equation, we can see that the target value is calculated by the next agent $i+1$. During a learning step, the parameters of each agent are updated sequentially. However, we reverse the sequence that users take actions, such that the parameters of agent $i+1$ is updated before, and can be used in (12) for agent i. For simplicity, we still use a fixed user sequence, so the training process of multiagent stepwise RL is similar to Algorithm 1. The differences are described below. First, at line 7, a is selected by using agent i's D3QN. Second, at line 11, the training sample is saved in agent i's memory buffer. Third, the learning process from lines 12 to 19

Authorized licensed use limited to: Nanjing University. Downloaded on April 09,2025 at 11:23:54 UTC from IEEE Xplore. Restrictions apply.

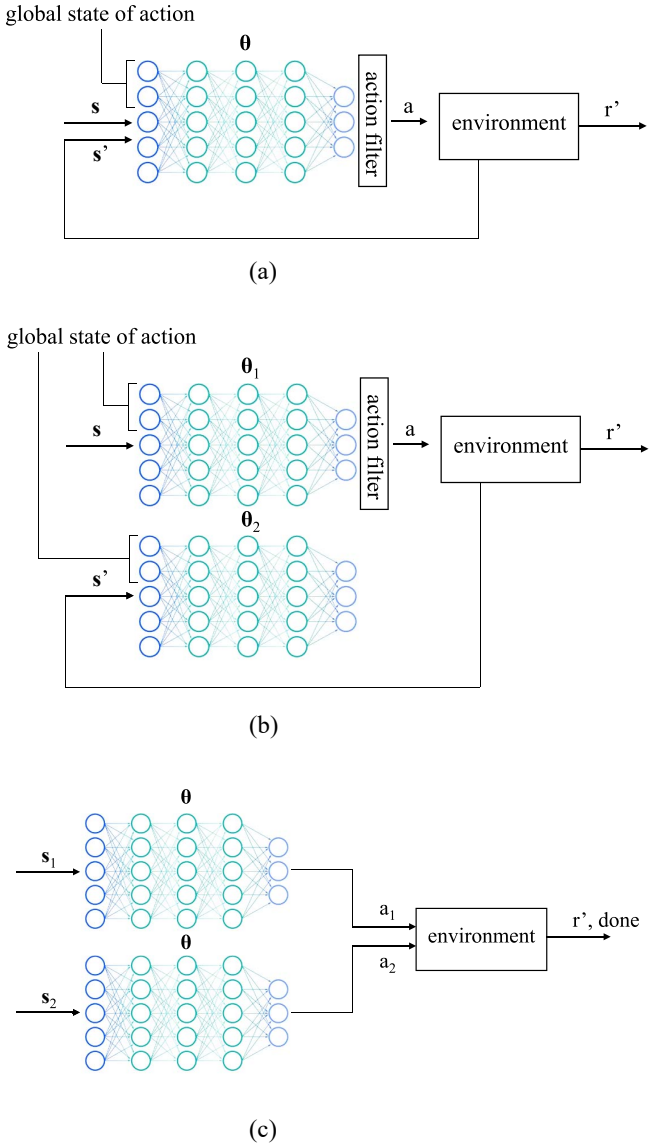


Fig. 2. Illustration of the (a) Single-agent stepwise RL, (b) Multiagent stepwise RL, and (c) Standard multiagent RL.

is conducted for all agents in the sequence of $N, \ldots, 1$, and the loss is calculated by (12). Besides, since there are more D3QNs to be trained, more episodes should be run, such that the number of training samples in the memory buffer of each agent is at the same level of single-agent case.

E. Summary of Different RL Methods

At the end of this section, we summarize and compare the three introduced RL methods through illustrative figures shown in Fig. 2. In single-agent stepwise RL [Fig. 2(a)], only one network parameterized by $\boldsymbol{\theta}$ is used. In each step, the action of a user is determined by the network, and the action is encoded into the state vector. Also, action filter is adopted such that only feasible actions can be selected. In multiagent stepwise RL [Fig. 2(b)], the main difference is that separate networks parameterized by $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots,$ are used for users. In each step, the action of a user is determined by its own network. In standard multiagent RL [Fig. 2(c)], different agent's networks use shared parameters $\boldsymbol{\theta}$, so there is actually only one network. All users' actions are determined by the network by inputting their own states $\mathbf{s}_1, \mathbf{s}_2, \ldots,$. The state encodes users' IDs so as to distinguish between different users. The actions are taken simultaneously in one step and a common reward is feedback. Also, *done* is feedback together with reward, which indicates that the episode ends.

From the above summary, the time and space complexity of different RL methods can be straightforwardly known. The time complexity of single-agent stepwise RL for calculating the subchannel allocation results is $\mathcal{O}(N)$, in terms of the number of forward propagations (namely, the inference process) through a single D3QN network. For both multiagent RL methods, the time complexity is constant. However, the space complexity of multiagent stepwise RL is $\mathcal{O}(N)$, in terms of the number of different D3QNs (namely, different sets of D3QN parameters). On the contrary, the space complexity of single-agent stepwise RL and standard multiagent RL (with parameter sharing technique employed) is constant.

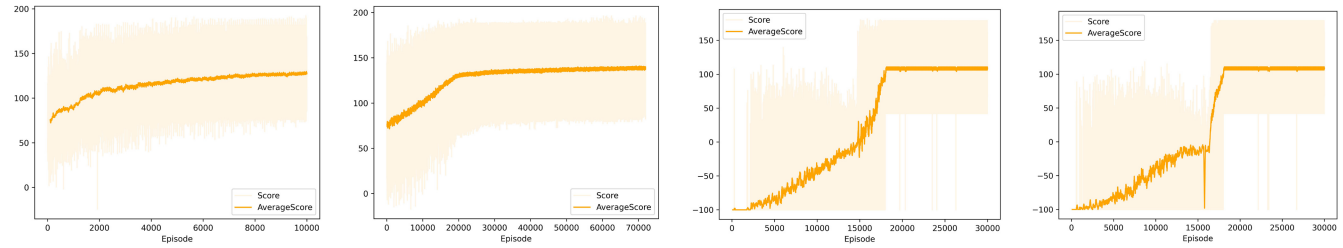
V. Simulation Results

A. Simulation Setting

We consider a network with one MBS and three nonoverlapping, equally separated SBSs surrounding the MBS. We totally generate 100 random environments with different seeds, 80 for training, and 20 for testing. Resource allocation results will be given and evaluated for each environment for all the algorithms. In each environment, $N_{\text{SBS}} = N/3$ users are generated at random locations within the coverage of each SBS such that each user is accessible to both the MBS as well as one SBS. Considering that the various channel effects will finally lead to SNR variations, the SNR values input to the MAC layer should be random such that the algorithms can be evaluated in a relatively practical communication environment. In the simulation, the SNR values of all links are derived from randomly generated user locations considering the large-scale channel fading effect. Note that although more channel effects, such as small-scale fading, can be further considered, it will essentially make no difference to the training and testing of RL algorithms, since from the perspective of MAC layer, the input SNR values are still random after superimposing the other random variations from more channel effects. Hence, the simulation can fit the actual communication scenario. In order to validate that our proposed stepwise RL methods can be scaled to more users, we train different D3QNs for $N = 6/9/12/15$ users and test their performance. We quantize the value of services and price of BSs to 10 levels $[1, \ldots, 10]$. Users' values of services are randomly generated within the range. The prices of BSs are set beforehand, and the price of MBS is higher than SBSs, since more users can access the MBS. The parameter settings used in the simulation are listed in Table II.

The neural network architecture of D3QN contains three layers, namely, the input layer and two fully connected layers with 128 neurons. All the values in state vector are normalized to $[0, 1]$ before inputting the neural network. For

Authorized licensed use limited to: Nanjing University. Downloaded on April 09,2025 at 11:23:54 UTC from IEEE Xplore. Restrictions apply.



(a) (b) (c) (d)

Fig. 3. Training processes of different learning methods for 12 users. (a) Single-agent stepwise RL. (b) MA. (c) MA-j. (d) MA-f.

TABLE II
PARAMETER SETTINGS

parameters	values
No. of users N	$\{6, 9, 12, 15\}$
No. of MBS's subchannels K^u_{MBS},K^d_{MBS}	4,4
No. of SBS's subchannels K^u_{SBS},K^d_{SBS}	4,4
subchannel bandwidth	1 MHz
MBS carrier frequency	2.9 GHz
SBS carrier frequency	3.5 GHz
MBS transmit power	46 dBm
SBS transmit power	20 dBm
UE transmit power	20 dBm
noise power spectrum density	-174 dBm/Hz
noise power on subchannel	-114 dBm
channel gain g	free-space path loss
MBS price p^u_{MBS},p^d_{MBS}	6,6
SBS price p^u_{SBS},p^d_{SBS}	4,4
rate demand r^u_{min},r^d_{min}	$[0.5, 2]$, $[0.5, 2]$ Mbps
uplink/downlink weight ω	0.5
utility function parameters α, β, λ	0.88,0.88,-2.25
utility function input scaling	about $[-10, 10]$

TABLE III
PARAMETER SETTINGS FOR LEARNING

parameters	values
activation function	ReLU
optimizer	Adam
learning rate	0.0003
reward discounting γ	0.99
ε initial value	0.1
ε linear decay rate	1e-4
memory buffer size	100000
target network update period τ	100
mini-batch size	64
training seeds	$[11, 90]$
testing seeds	$[1, 10]$, $[91, 100]$
No. of episodes for *SA*	10000/15000
No. of episodes for *MA*	$6000N$
No. of episodes for *MA-j*	30000
Reward penalty for *MA-j*	-100

the three different algorithms introduced in Section IV-B, Section IV-D, and Section IV-C (denoted by SA, MA and MA-j, respectively), we run different number of episodes during training process. We also evaluate the multiagent RL algorithm in [26] (denoted by MA-f), which adopts a fingerprint-based approach [28] to stabilize training and address the nonstationary environment problem. The algorithm is implemented based on MA-j, while adding the current training episode and ε value into the state, as described by [26]. The trained networks are then used to for different environments generated from the test seeds. The parameter settings relevant to the learning process are listed in Table III. As the baseline without utilizing learning and UCRA, a greedy algorithm (denoted by *SNR-g*) that allocates subchannels to users based on the SNR values is further given. For the uplink/downlink of a user i, the MBS or SBS with higher SNR is selected, and $\chi_{i,j} = \min\{2, \bar{K}_j\}$ subchannels are allocated, where $\bar{K}_j$ is the remaining subchannels of the selected BS j. If $\bar{K}_j = 0$, then the other BS j' is selected. If $\bar{K}_{j'} = 0$ for either uplink or downlink, then the user is not allocated any subchannels.

B. Results

We first show the training processes of different learning methods, namely, SA, MA, MA-j, and MA-f in Fig. 3. We use the case of 12 users as examples, and plot the score of each training episode, which is actually the sum of reward of each step in the episode. From our definition of reward and the stepwise RL process, each episode corresponds to an environment, and the score is the objective of our subchannel allocation problem, namely, the sum, utility of all users in the environment. In order to observe the trend more clearly, we also plot the score averaged over a window of the last 100 episodes. From Fig. 3, we can see that the average scores of all the four methods become steady after a certain number of training episodes. This indicates that the subchannel allocation results obtained from these methods for each environment become stable. In other words, the parameters of neural networks have converged (no further updates even with more training episodes), and the neural networks have completed training. In our simulation, the number of episodes to reach convergence is near 20000 for MA-j. On the one hand, this value is usually quite different for different RL methods in literatures, since the underlying environment settings, the complexity of problem, and the definition of training episodes are very distinct. For example, about 200000 training steps are required to converge in [20]. On the other hand, compared to SA, the nonstationary environment problem in MA-j makes the training process more unstable and difficult to converge. Further, MA is the most stable method, and achieves the highest average score when the training curve becomes steady. As mentioned earlier, the main advantage of MA compared to SA is that it utilizes a separate neural network for each user. Both MA and SA do not have the nonstationary environment problem, so they perform better than MA-j and MA-f. Another

Authorized licensed use limited to: Nanjing University. Downloaded on April 09,2025 at 11:23:54 UTC from IEEE Xplore. Restrictions apply.

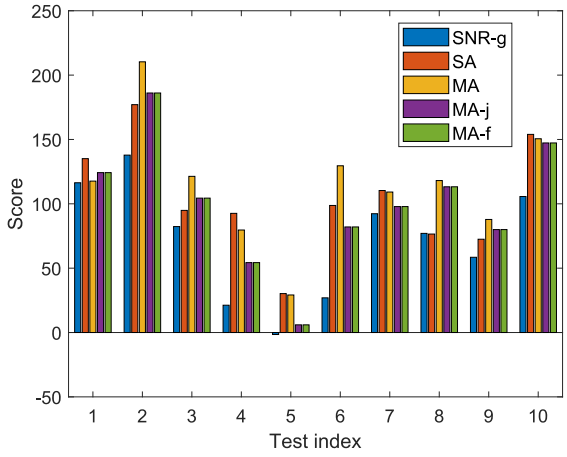


Fig. 4. Scores of ten test environments for 12 users.

reason is the action filter, which can only be used in MA and SA due to the stepwise process. The advantage of action filter is clearly observed from the beginning episodes, where the scores of MA-j and MA-f start from -100, indicating the reward penalty due to impossible joint actions that lead to more allocated subchannels than the total number.

To further demonstrate the performance improvement of proposed algorithms, we show the scores of ten test environments (generated from seeds $[1, \ldots, 10]$) of SA, MA, MA-j, MA-f and SNR-g in Fig. 4. First of all, we can find that the scores of different test environments vary dramatically, due to the randomness of generated environment. This phenomenon corresponds to the oscillations observed in the score of each episode in Fig. 3, since the score of each episode is determined by the randomly generated environment. It is clear that the performance of all the learning-based algorithms is better than the baseline SNR-g for all test environments. Notably, in tests #4, #5, #6, SA and MA are much better than SNR-g, MA-j and MA-f, which means that they are capable of finding better subchannel allocation schemes in these environments. Also, MA has much higher scores than SA in tests #2, #3, #6, #8, #9, while SA is much better than MA in tests #1, #4. Thus, MA can generally outperform SA, but it is obviously not impossible for all test cases. We can further observe that the results of MA-j and MA-f are the same for these test environments (as well as others, such as seeds $[91, \ldots, 100]$), indicating that they derive the same subchannel allocation decisions. We conclude that the fingerprint method adopted by MA-f does not lead to significant changes to MA-j after the neural network is fully trained (also can be observed from the training process in Fig. 3), therefore the nonstationary environment problem is still not well solved by MA-f. Intuitively, the performance of both MA-f and MA-j will not be comparable to our proposed methods. In the following comparison results, MA-f will not be shown in the figures for clarity.

In order to make a thorough comparison among these methods, we further test 20 environments (generated from seeds $[1, \ldots, 10]$ and $[91, \ldots, 100]$) for different number of users ($N = 6, 9, 12, 15$). The results are given from Figs. 5–8. Specifically, we plot the kernel density estimate

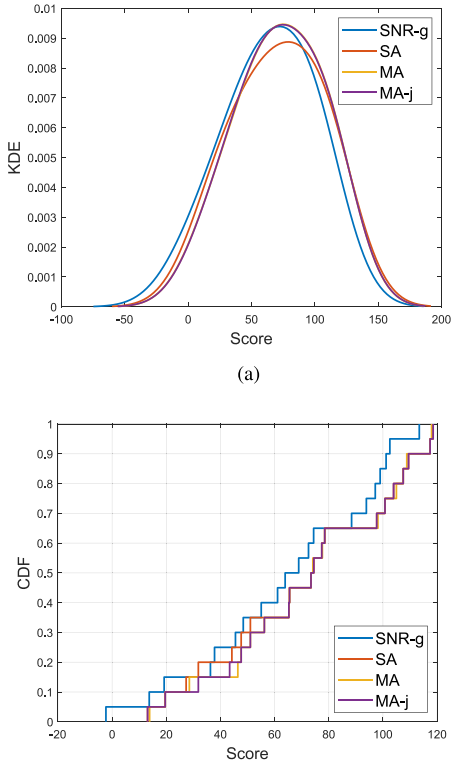


Fig. 5. Comparison of different methods for six users. (a) KDE. (b) CDF.

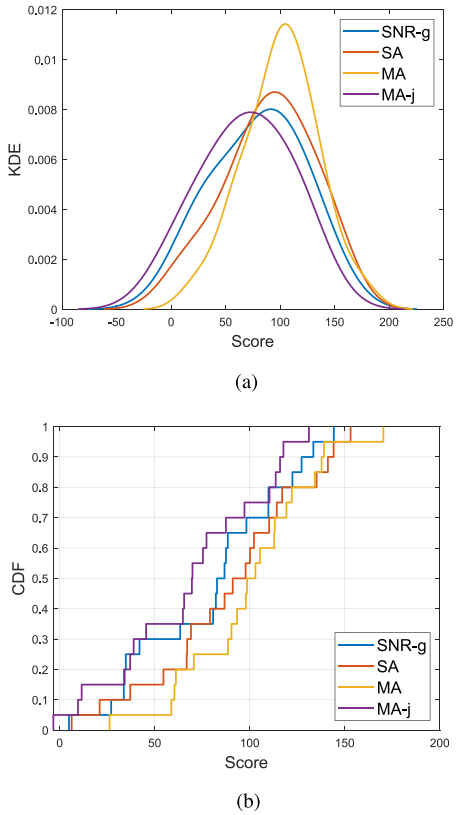


Fig. 6. Comparison of different methods for nine users. (a) KDE. (b) CDF.

(KDE) and cumulative distribution function (CDF) of scores from these test environments for different methods. KDE is an estimation of probability distribution function (PDF) with

Authorized licensed use limited to: Nanjing University. Downloaded on April 09,2025 at 11:23:54 UTC from IEEE Xplore. Restrictions apply.

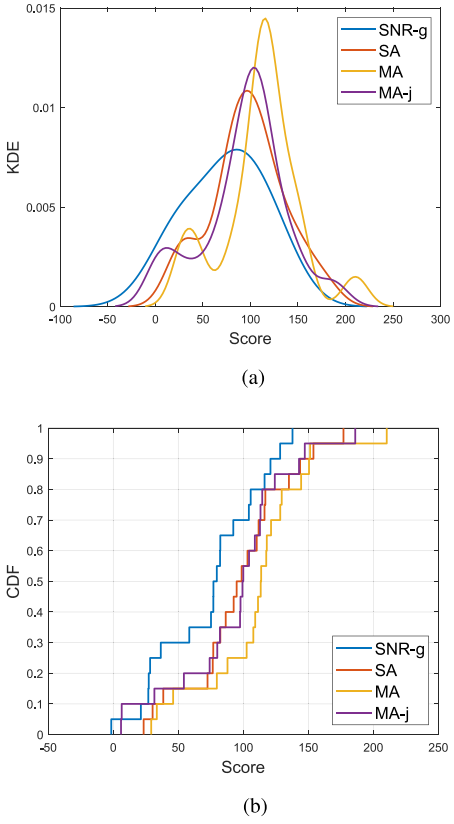


Fig. 7. Comparison of different methods for 12 users. (a) KDE. (b) CDF.

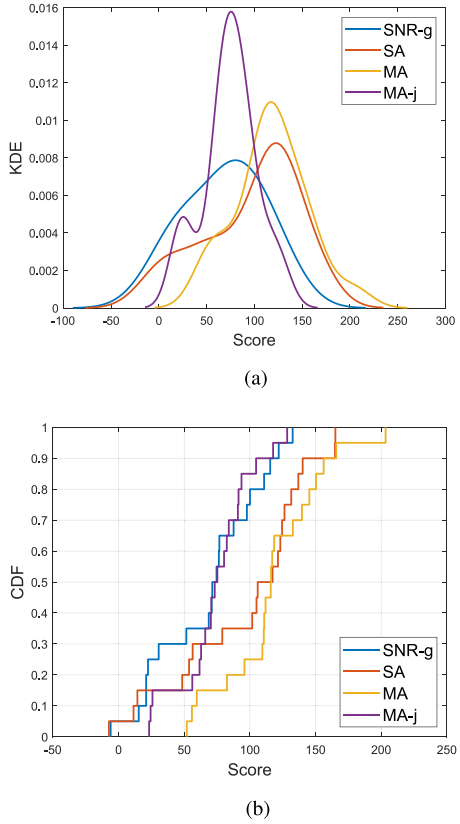


Fig. 8. Comparison of different methods for 15 users. (a) KDE. (b) CDF.

underlying data (i.e., the scores from test environments). By using KDE and CDF, the distribution of scores can be visually shown. From these figures, we have the following observations. First, the SA and MA learning methods perform better than the baseline method SNR-g and the standard multiagent RL method MA-j, thereby demonstrating the superiority of these two methods. Second, when the number of users gets larger, SA and MA can still work well. Furthermore, the performance gap between SA/MA and MA-j becomes larger, and the performance of MA-j is also not stable. The main reason is that the nonstationary environment problem becomes more influential in MA-j. For example, in Fig. 6, we can see that even SNR-g is better than MA-j. Last, when the number of users is small, e.g., the 6 users case in Fig. 5, the learning-based methods can still find out better subchannel allocation solutions, given that subchannels are sufficient and all users can be allocated subchannels. We should clarify that the performance improvement of SA/MA against comparative methods may seem not that huge, and it is because in most cases there are almost no subchannels left after the allocation.

VI. Conclusion

In this article, we have studied UCRA in FD-RAN with an underlying HetNet, such that the flexibility of resource allocation is exploited for realizing personalized service provision to users. We have designed a new user utility function by specifically considering users' different subjective values on services. A subchannel allocation problem was formulated, and a stepwise RL method was designed to solve it. We have also extended the method to multiagent case. Various techniques were employed to make the proposed RL methods more stable and efficient, and suitable for more users/agents in multiagent case (without having to face the nonstationary environment problem). The framework of our proposed stepwise RL approach can be used to general resource allocation problems with other objective functions and constraints. It can also be adapted to other kind of network topologies. Furthermore, our defined user utility is not restricted to specific user requirements. Our future work will consider the core network and protocol/interface design for UCRA in FD-RAN.

References

[1] Q. Yu et al., "A fully-decoupled RAN architecture for 6G inspired by neurotransmission," *J. Commun. Inf. Netw.*, vol. 4, no. 4, pp. 15–23, Dec. 2019.

[2] B. Qian et al., "Enabling fully-decoupled radio access with elastic resource allocation," *IEEE Trans. Cogn. Commun. Netw.*, vol. 9, no. 4, pp. 1025–1040, Aug. 2023.

[3] Y. Lin, W. Bao, W. Yu, and B. Liang, "Optimizing user association and spectrum allocation in HetNets: A utility perspective," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 6, pp. 1025–1039, Jun. 2015.

[4] N. C. Luong, P. Wang, D. Niyato, Y.-C. Liang, Z. Han, and F. Hou, "Applications of economic and pricing models for resource management in 5G wireless networks: A survey," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 4, pp. 3298–3339, 4th Quart., 2018.

[5] H. U. Sokun, R. H. Gohary, and H. Yanikomeroglu, "A novel approach for QoS-aware joint user association, resource block and discrete power allocation in HetNets," *IEEE Trans. Wireless Commun.*, vol. 16, no. 11, pp. 7603–7618, Nov. 2017.

Authorized licensed use limited to: Nanjing University. Downloaded on April 09,2025 at 11:23:54 UTC from IEEE Xplore. Restrictions apply.

[6] N. Tadayon and S. Aissa, "Radio resource allocation and pricing: Auction-based design and applications," *IEEE Trans. Signal Process.*, vol. 66, no. 20, pp. 5240–5254, Oct. 2018.
[7] J. Chen, B. Qian, Y. Xu, H. Zhou, and X. S. Shen, "Towards user-centric resource allocation for 6G: An economic perspective," *IEEE Netw.*, vol. 37, no. 2, pp. 254–261, Mar./Apr. 2023.
[8] Y. Yang et al., "6G network AI architecture for everyone-centric customized services," *IEEE Netw.*, vol. 37, no. 5, pp. 71–80, Sep. 2023.
[9] J. S. Levy, "An introduction to prospect theory," *Political Psychol.*, vol. 13, no. 2, pp. 171–186, 1992.
[10] Z. Sattar, J. V. C. Evangelista, G. Kaddoum, and N. Batani, "Spectral efficiency analysis of the decoupled access for downlink and uplink in two-tier network," *IEEE Trans. Veh. Technol.*, vol. 68, no. 5, pp. 4871–4883, May 2019.
[11] L. Zhang, W. Nie, G. Feng, F.-C. Zheng, and S. Qin, "Uplink performance improvement by decoupling uplink/downlink access in HetNets," *IEEE Trans. Veh. Technol.*, vol. 66, no. 8, pp. 6862–6876, Aug. 2017.
[12] C. He, Y. Hu, Y. Chen, and B. Zeng, "Joint power allocation and channel assignment for NOMA with deep reinforcement learning," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 10, pp. 2200–2210, Oct. 2019.
[13] K. Yu, H. Zhou, Z. Tang, X. Shen, and F. Hou, "Deep reinforcement learning-based RAN slicing for UL/DL decoupled cellular V2X," *IEEE Trans. Wireless Commun.*, vol. 21, no. 5, pp. 3523–3535, May 2022.
[14] O. Naparstek and K. Cohen, "Deep multi-user reinforcement learning for distributed dynamic spectrum access," *IEEE Trans. Wireless Commun.*, vol. 18, no. 1, pp. 310–323, Jan. 2019.
[15] S. Wang, H. Liu, P. H. Gomes, and B. Krishnamachari, "Deep reinforcement learning for dynamic multichannel access in wireless networks," *IEEE Trans. Cogn. Commun. Netw.*, vol. 4, no. 2, pp. 257–265, Jun. 2018.
[16] L. Tan, Z. Zhu, F. Ge, and N. Xiong, "Utility maximization resource allocation in wireless networks: Methods and algorithms," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 45, no. 7, pp. 1018–1034, Jul. 2015.
[17] S. Basu, S. Roy, S. Bandyopadhyay, and S. D. Bit, "A utility driven post disaster emergency resource allocation system using DTN," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 50, no. 7, pp. 2338–2350, Jul. 2020.
[18] X. Shen, J. Gao, W. Wu, M. Li, C. Zhou, and W. Zhuang, "Holistic network virtualization and pervasive network intelligence for 6G," *IEEE Commun. Surveys Tuts.*, vol. 24, no. 1, pp. 1–30, 1st Quart., 2022.
[19] W. Wu et al., "Split learning over wireless networks: Parallel design and resource management," *IEEE J. Sel. Areas Commun.*, vol. 41, no. 4, pp. 1051–1066, Apr. 2023.
[20] H.-H. Chang, H. Song, Y. Yi, J. Zhang, H. He, and L. Liu, "Distributive dynamic spectrum access through deep reinforcement learning: A reservoir computing-based approach," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1938–1948, Apr. 2019.
[21] J. Liu, X. Tao, and J. Lu, "Mobility-aware centralized reinforcement learning for dynamic resource allocation in HetNets," in *Proc. IEEE Glob. Commun. Conf. (GLOBECOM)*, 2019, pp. 1–6.
[22] Y. Wei, F. R. Yu, M. Song, and Z. Han, "User scheduling and resource allocation in HetNets with hybrid energy supply: An actor–critic reinforcement learning approach," *IEEE Trans. Wireless Commun.*, vol. 17, no. 1, pp. 680–692, Jan. 2018.
[23] J. Tan, Y.-C. Liang, L. Zhang, and G. Feng, "Deep reinforcement learning for joint channel selection and power control in D2D networks," *IEEE Trans. Wireless Commun.*, vol. 20, no. 2, pp. 1363–1378, Feb. 2021.
[24] Z. Li and C. Guo, "Multi-agent deep reinforcement learning based spectrum allocation for D2D underlay communications," *IEEE Trans. Veh. Technol.*, vol. 69, no. 2, pp. 1828–1840, Feb. 2020.
[25] H. Ye, G. Y. Li, and B.-H. F. Juang, "Deep reinforcement learning based resource allocation for V2V communications," *IEEE Trans. Veh. Technol.*, vol. 68, no. 4, pp. 3163–3173, Apr. 2019.
[26] L. Liang, H. Ye, and G. Y. Li, "Spectrum sharing in vehicular networks based on multi-agent reinforcement learning," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 10, pp. 2282–2292, Oct. 2019.
[27] P. Xiang, H. Shan, M. Wang, Z. Xiang, and Z. Zhu, "Multi-agent RL enables decentralized spectrum access in vehicular networks," *IEEE Trans. Veh. Technol.*, vol. 70, no. 10, pp. 10750–10762, Oct. 2021.
[28] J. Foerster et al., "Stabilising experience replay for deep multi-agent reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 1146–1155.
[29] S. Sen, C. Joe-Wong, S. Ha, and M. Chiang, "Smart data pricing: Using economics to manage network congestion," *Commun. ACM*, vol. 58, no. 12, pp. 86–93, 2015.
[30] J. Liu, J. Chen, C. He, and H. Zhou, "Leveraging load-aware dynamic pricing for cell-level demand-supply equilibrium," *IEEE Trans. Veh. Technol.*, vol. 72, no. 5, pp. 6902–6906, May 2023.
[31] Q. Yu, J. Ren, H. Zhou, and W. Zhang, "A cybertwin based network architecture for 6G," in *Proc. 2nd 6G Wireless Summit (6G SUMMIT)*, 2020, pp. 1–5.
[32] T. T. Nguyen, N. D. Nguyen, and S. Nahavandi, "Deep reinforcement learning for multiagent systems: A review of challenges, solutions, and applications," *IEEE Trans. Cybern.*, vol. 50, no. 9, pp. 3826–3839, Sep. 2020.
[33] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 2018.
[34] V. Mnih et al., "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.
[35] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in *Proc. AAAI Conf. Artif. Intell.*, 2016, pp. 2094–2100.
[36] Z. Wang, T. Schaul, M. Hessel, H. Hasselt, M. Lanctot, and N. Freitas, "Dueling network architectures for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2016, pp. 1995–2003.

Jiacheng Chen (Member, IEEE) received the Ph.D. degree in information and communications engineering from Shanghai Jiao Tong University, Shanghai, China, in 2018.

From 2015 to 2016, he was a Visiting Scholar with BBCR Group, University of Waterloo, Waterloo, ON, Canada. He is currently an Assistant Researcher with Peng Cheng Laboratory, Shenzhen, China. His research interests include future network design, AI-enabled 6G network, and resource management.

Dr. Chen has won the JCIN Best Paper Award in 2016, the Chinese Institute of Electronics Outstanding Scientific Paper in the Field of Electronic Information in 2020, and the IEEE PIMRC'23 Best Paper Award. He has served as the Guest Editor for IEEE INTERNET OF THINGS JOURNAL and *Journal of Communications and Information Networks* (JCIN), and the Workshop Co-Chair for IEEE/CIC ICCC from 2021 to 2023.

Jingbo Liu received the B.S. degree in communication engineering from Nanjing University of Posts and Telecommunications, Nanjing, China, in 2020. He is currently pursuing the Ph.D. degree with the School of Electronic Information and Electrical Engineering, Shanghai Jiao Tong University, Shanghai, China.

He is also an Intern with Peng Cheng Laboratory, Shenzhen, China. His current research interests include resource management in wireless networks, reinforcement learning, and fully-decoupled radio access network.

Haibo Zhou (Senior Member, IEEE) received the Ph.D. degree in information and communication engineering from Shanghai Jiao Tong University, Shanghai, China, in 2014.

From 2014 to 2017, he was a Postdoctoral Fellow with the Broadband Communications Research Group, Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON, Canada. He is currently a Full Professor with the School of Electronic Science and Engineering, Nanjing University, Nanjing, China. His research interests include resource management and protocol design in B5G/6G networks, vehicular ad hoc networks, and space–air–ground integrated networks.

Prof. Zhou was a recipient of the 2019 IEEE ComSoc Asia–Pacific Outstanding Young Researcher Award, the 2023 IEEE ComSoc WTC Young Researcher Award, the 2023–2024 IEEE ComSoc Distinguished Lecturer, and the 2023–2025 IEEE VTS Distinguished Lecturer. He served as a Track/Symposium Co-Chair for IEEE/CIC ICCC 2019, IEEE VTC-Fall 2020, IEEE VTC-Fall 2021, WCSP 2022, IEEE GLOBECOM 2022, IEEE/CIC ICCC 2024, IEEE ICC 2024, and IEEE GLOBECOM 2024. He is currently an Associate Editor of the IEEE TRANSACTIONS ON WIRELESS COMMUNICATIONS, IEEE INTERNET OF THINGS JOURNAL, *IEEE Network Magazine*, and *Journal of Communications and Information Networks*.

Authorized licensed use limited to: Nanjing University. Downloaded on April 09,2025 at 11:23:54 UTC from IEEE Xplore. Restrictions apply.