Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 9391939, 13 pages
https://doi.org/10.1155/2022/9391939

Research Article

Intelligent Dynamic Spectrum Allocation in MEC-Enabled Cognitive Networks: A Multiagent Reinforcement Learning Approach

Chan Lei, Haitao Zhao, Li Zhou, Jiao Zhang, Haijun Wang, and Haitao Chen

College of Electronic Science and Technology, National University of Defense Technology, Changsha 410073, China

Correspondence should be addressed to Haitao Zhao; haitaozhao@nudt.edu.cn

Received 22 June 2022; Revised 17 August 2022; Accepted 25 August 2022; Published 25 September 2022

Academic Editor: Amr Tolba

Making effective use of scarce spectrum resources, along with efficient computational performance, is one of the key challenges for future wireless networks. To tackle this issue, in this paper, we focus on the intelligent dynamic spectrum allocation (DSA) in a mobile edge computing (MEC) enabled cognitive network. And our objective is to optimize the spectrum utilization and load balance among idle channels. Since users can only acquire part of environment information in a decentralized way, we model such a problem as decentralized partially observed Markov decision process (Dec-POMDP) and design the corresponding evaluating metric to encourage users sense and access spectrum properly. Then, we propose a QMIX-based DSA method with centralized training decentralized execution (CTDE) structure to tackle it. In the training phase, the users offload the computational tasks to the MEC server to obtain the optimal distributed DSA strategies, through which the users select the optimal channel locally in the execution phase. Simulation results show that, using the proposed algorithm, users can independently capture spectrum holes, and hence improve the spectrum utilization while balancing the load on available channels.

1. Introduction

With the rapid development of the future beyond 5G/6G wireless communication, more and more emerging applications like virtual reality (VR), augmented reality (AR), and interactive game are springing up, requiring low-latency, high-reliability and powerful computational capability [1]. Due to size, weight, and power (SWaP) constraints, devices are recognized to have limited computational capability to fully support these applications. Mobile edge computing (MEC) has a great potential to overcome this issue, through which the users can offload the computational tasks to MEC servers [2–4]. By this way, many services can be provided, like communication, caching, and computing [5]. Meanwhile, it can also potentially boost intelligence for users to achieve efficient spectrum access with low coordination overhead. Confronting the time-varying and complex wireless environment, it is challenging for users to access the spectrum adaptively with robustness guarantee. The MEC-enabled intelligent dynamic spectrum allocation (DSA) can play a significant role in this case.

Many efforts have been devoted on the DSA in the traditional way, e.g., blind rendezvous in D2D [6] and cognitive radio networks (CRNs) [7], cross-layer perspective [8, 9] and randomized rounding algorithm [10] in CRNs, and bipartite graph theory in wireless LANs [11]. Since these works require lots of statistical knowledge, which is difficult to obtain in a dynamic network, the intelligent approaches have been adopted for DSA further. In [12], genetic algorithm (GA) is adopted by the central node to complete DSA for each secondary user (SU) in a CRN. Benefit from the fitting properties of deep neural networks and the interaction with the environment, several works focus on the deep reinforcement learning (DRL) structure [13]. Specially, in [14, 15], the central controller is employed to evaluate and allocate channels to multiuser through DRL, while the users report the channel state after having an access. And the maximum channel utilization and minimum collision are

expected to be optimized with no prior information for multiuser. However, there is still much room for improvement compared with the optimal scheme, due to the heavy signalling overhead and overdependence on the central node. To tackle this problem, there are increasing works that focus on intelligent DSA in a distributed way. In [16], the bioinspired solution is employed for users to adjust and optimize the cluster size distributedly in cognitive internet of things (IoT), which aims to achieve efficient spectrum allocation, flexible connection, and minimum network access delay. In [17], a heuristic method is applied for users to sense an unoccupied spectrum and build an optimum route, so as to complete dynamic spectrum allocation and power control. The authors in [18] have modeled the multiuser multichannel allocation as an undirected graph, and a greedy algorithm is designed to form the load balanced cluster. The authors in [19, 20] introduce the game theory for SUs to allocate spectrum, and the SUs learn access strategies competitively by maximizing their respective revenue. The authors in [21] have investigated the problem of multiuser enabled frequency division multiple access in radio network, and a bargaining approach is used to allocate subcarrier and power for multiuser by get a Nash equilibrium, so as to achieve the tradeoff between throughput and power consumption. Even though these works make a good breakthrough in the intelligent DSA, there are still some shortcomings. For one thing, these methods are just feasible for the simple case with small strategy space, otherwise, the long decision time would occur. Obviously, it is impractical for the low-latency scenarios. For another, the interaction should be addressed in the dynamic environment furtherly.

To tackle the dilemmas above, a powerful method, mutliagent reinforcement learning (MARL), has gained increasing interest recently. MARL is an extension of reinforcement learning (RL) [22], which is suitable for distributed learning and processing. With the aid of this method, users can interact with environment and obtain their DSA strategies as agents. Particularly, in [23, 24], the double deep Q-network (DQN) algorithm and multiagent Q-learning are applied to active users, who compete to access multichannel independently to achieve minimum collisions. In [25], the SUs in CRN are aiming to learn proper spectrum access strategies autonomously and the DRL method, along with echo state network (ESN) is adopted, through which interference can be alleviated. And in [26], a competitive spectrum access scheme for multiuser is proposed, and the performance of DSA and avoidance is simply analyzed by Q-learning method. While in [27], multiuser DSA is modeled as a multiarmed bandit problem, in which the users are supposed to access proper spectrum distributedly. However, since there is no negotiation that users analyze information locally and allocate the spectrum resources by way of competition in these works, the spectrum access can be regarded as an ALOHA-like process, which is still confronted with great challenges in highly dynamic environments. Moreover, for constraints of computing and battery capacity, there is still a huge gap to fill before deployment in real world.

Inspired by MEC technology, which could extend the computational capacity and process the task for DSA at edge cloud platform [28–30], the centralized training decentralized execution (CTDE) scheme of MARL is addressed to achieve efficient DSA [31–33]. By this way, the distributed users offload the DSA task to the cloud platform in the training phase, so as to learn the action-taking strategies, and then work in a fully distributed manner in the practical implementation phase. Specially, the authors in [31] have investigated a noncooperative DSA in CRN. In [32], users are expected to transmit on idle channel and cooperate to maximize the sum rate, and the double DQN algorithm is utilized, resulting in a certain gap with the optimal performance. Considering the linear relationship between the global and individual utility, the authors in [33] propose a QMIX-based DSA algorithm, which brings the excellent strategies for users, and the maximum successful transmission and minimum collision are realized. However, in order to avoid collision, the users in [32, 33] are allowed to be silent. That is, it is assumed that only part of users could participate in the DSA at the same time, which is unfair to all users distributed in the network. Moreover, the works in [18, 24, 27] take the perfect spectrum detection into consideration, where the selection for idle spectrum is always guaranteed. In reality, influenced by the dynamic environment and limited hardware condition, the detection ability is usually partial [34] and imperfect [35].

In this paper, we investigate the intelligent DSA for multiuser MEC networks. We consider that the users can only sense part of information, and the ability of detection is assumed to be imperfect. Motivated by the monotonicity of the considered problem, as well as the powerful computation ability, the QMIX-based DSA algorithm with CTDE structure is employed. To the best of our knowledge, this work has not been researched yet. And we highlight the main contributions of our work as follows:

(1) We focus on a MEC-enabled cognitive network deploying multiple SUs who attempt to access the dynamic spectrum without perfect sensing capabilities. In this scenario, SUs are supposed to be intelligent to achieve their common task autonomously. Meanwhile, in order to make up for the users' limited computational capability and energy reserve, the MEC server, which is computationally powerful and long-lived, is employed at BS. This is practically significant, since the traditional dependence on a central controller is released. And the users can adapt to the environment independently and timely with lower overhead, so that it can be further extended to the latency-sensitive applications

(2) We formulate a distributed DSA problem to improve both the idle channel utilization and load balance. And the problem is modeled as a decentralized partial observation Markov decision process (Dec-POMDP). Then we propose a CTDE enabled DSA algorithm, whose characteristic is consistent with that of the modeled problem. This algorithm can handle the environment dynamics and users' partial observation with low-complexity for practical implementation. Specially, different from these online searching methods, such as POMCP [36], DESPOT [37], and HyP-DESPOT [38], there are two phases

in our proposed algorithm, i.e., offline training and online execution. In the offline phase, the task of DSA is offloaded to the MEC. With the aid of MEC, the SUs adaptively adjust their DSA strategies, so that their network models can be well trained finally. While in the online phase, each SU executes action locally based on the trained model with no central controller and the coordination among SUs

(3) We present simulations to demonstrate the effectiveness and feasibility of our proposed DSA algorithm in the different settings under dynamic environment. We observed that the optimal network utility is always realized after limited training, while the sensing accuracy is also improved. The SUs can effectively overcome their imperfect sensing characteristic and capture the idle channels. Based on this, the expected optimal DSA task could be completed in a fully distributed manner

The rest of this paper is structured as follows: the system model is provided in Section 2. In Section 3, the problem is formulated as a Dec-POMDP to maximize the global utility. Then in Section 4, QMIX-based algorithm is proposed to obtain the optimal DSA policy. Numerical results are provided in Section 5, and the conclusion of the paper is presented in Section 6.

2. System Model

As shown in Figure 1, we consider a MEC-enabled cognitive network consisting of one primary user (PU), N SUs, with the set denote by $\mathrm{SN} = \{\mathrm{SU}_1, \mathrm{SU}_2, \cdots, \mathrm{SU}_N\}$, and one cognitive BS with MEC server. There are M orthogonal authorized channels, denoted as $CH = \{\mathrm{ch}_1, \mathrm{ch}_2, \cdots, \mathrm{ch}_M\}$. The channels' state switches between idle and occupied according to the communication behavior of the PU. However, the channel state and switching pattern is unknown for SUs. We assume that there are $K(1 < K < M)$ idle channels, which are feasible for SUs distributed in the network to utilize opportunistically. In this paper, the SUs are supposed to capture the PUs occupation mode and learn to sense and access channels autonomously, so as to achieve the efficient DSA. Due to the limited computational ability and battery life, SUs offload their DSA tasks to the MEC server for computation and analysis. Thereafter, the MEC server distributes the DSA strategies to each SU for the online learning to realize the DSA. In the whole process, no information interaction is required among SUs.

We assume that all the SUs are slot-synchronized, and only part of primary channels can be sensed by each of them, one of which should be accessed by each SU further. Here, the energy detection mechanism is employed [39]. In practice, there are imperfect detections, which may cause the wrong judgement inevitably. And also, since the environment is unstable and channel states are time-varying as mentioned above, the SUs interact with environment to learn how to sense and access a particular channel.

Specially, as depicted in Figure 2, the whole DSA procedure for all SUs can be illustrated as follows: the PU occupies one or more channels at each time slot, and the state of primary channels may change at each time slot. Firstly, each SU

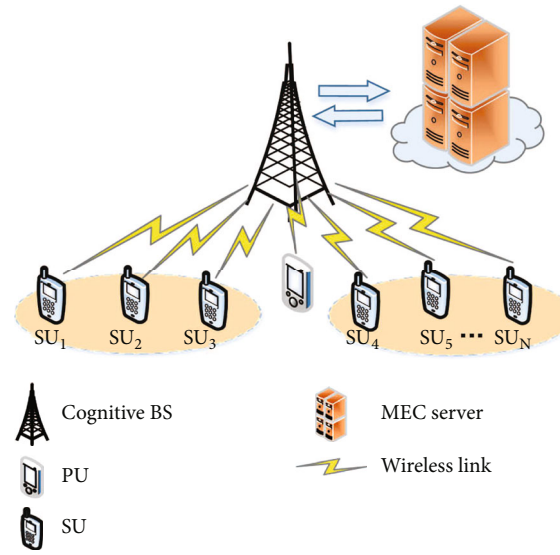

Figure 1: System model of the MEC-enabled cognitive network.

senses channels independently to judge whether they are occupied by the PU. Then, the SUs attempt to access one of the sensed channels and send the request signals to the BS, where the MEC server is employed to finish the computation and analysis so that the distributed DSA strategies for each users can be produced. In this way, SUs can learn the switching patterns of the channel states and decide which channel should be sensed in the next time, by analyzing their current DSA scheme and the corresponding feedback received from the BS.

3. Problem Definition with Dec-POMDP

Note that all SUs aim to achieve DSA, in which the objective including full idle channel utilization, and load balance is considered. For each SU, it can only sense part of primary channels, then judges the occupation state of the channel that it accesses without prior coordination among SUs. That is, the multiple SUs distributed in the network can only obtain partial environment information. Therefore, the problem can be modeled as Dec-POMDP, which can be formulated as a tuple $\langle N, S, O, A, P, r\rangle$. The definitions of the tuple elements are listed as follows, and some of the key symbols are summarized in Table 1.

N is the number of SUs who are regarded as multiple agents in the interactive environment.

"S is the global channel state space, which reflects the true state of M orthogonal authorized channels in the communication environment." At time slot t, the channel state space is defined as $S^t \triangleq \{s_1^t, \cdots, s_M^t\}$, where the state of ch_m, $m \in [1, M]$ is given by

$$s_m^t = \begin{cases} -1, \text{ if } ch_m, \text{ is busy at slot } t, \\ 1, \text{ if } ch_m, \text{ is idle at slot } t. \end{cases} \quad (1)$$

O is the partial observation space, and it represents the sensed channels for all agents. At each time slot t, agents

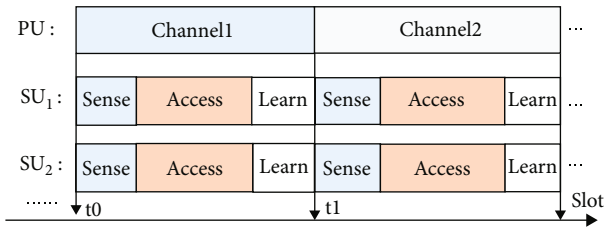

Figure 2: The time-block structure for the DSA procedure.

Table 1: Glossary of key symbols.

Symbol	Description
N	Number of agent
M	Number of orthogonal authorized channel
S	Global channel state space
S^t	Channel state space at slot t
s_m^t	State of channel m at slot t
O	Observation space
O^t	Observation space at slot t for all agents
o_n^t	Observation for agent n at slot t
$o_{n,m}^t$	Observation of channel m for agent n at slot t
A	Action space for all agents
A^t	Action profile for all agents at slot t
a_n^t	Actions for agent n at slot t
$a_{n,m}^t$	Action for agent n who chooses channel m at slot t
r	The set of immediate reward for all agents
r_n^t	Immediate reward for agent n at slot t
c_m^t	Number of agents allocated on the same channel m at slot t
r_{tot}^t	Total reward for all agents at slot t
τ_n	Observation-action history of agent n
R_{tot}	Global reward of all agents in the finite time slots
γ	Discount factor

observe some representation of environment state $O^t \triangleq \{o_1^t, \cdots, o_N^t\}$ from the state S^t. Particularly, the agent may not obtain full and perfect knowledge of the channel states, i.e., for agent $n \in [1, N]$, $o_n^t \neq S^t$. And we define the observation of the agent $n \in [1, N]$ for $ch_m (m \in [1, M])$ as $o_n^t = \{o_{n,m}^t | m \in [1, M]\}$, where

$$o_{n,m}^t = \begin{cases} -1, \text{ if } ch_m, \text{ is sensed busy at slot } t, \\ 1, \text{ if } ch_m, \text{ is sensed idle at slot } t. \end{cases} \tag{2}$$

A is the action space for all agents. The action profile for all agents at slot t is formulated as $A^t \triangleq \{a_1^t, \cdots, a_N^t\}$. For the agent $n \in [1, N]$ who chooses the ch_m, $m \in [1, M]$, we define $a_n^t = \{a_{n,m}^t | m \in [1, M]\}$, where

$$a_{n,m}^t = \begin{cases} -1, \text{ if the chosen } ch_m, \text{ is busy at slot } t, \\ 1, \text{ if the chosen } ch_m, \text{ is idle at slot } t. \end{cases} \tag{3}$$

P is the state transition matrix, reflecting the transition of channel occupation from state S^t to S^{t+1}.

$r = \{r_1, \cdots, r_N\}$ is the set of immediate reward for all agents after accessing the sensed channels, which encourages agents to learn an optimal DSA strategy. Here, the agents are supposed to independently sense and access a truly vacant and proper channel and obtain a reward according to the feedback from the BS.

The key point of the reward for each agent is to make perfect use of the idle channels in the space as fair as possible. Since the MEC server at BS will collect the channel state and the number of agents who request for the same channel, we design the immediate reward of the n-th agent at slot t as

$$r_n^t = \begin{cases} f\left(\frac{c_m^t}{N}\right), s_m^t = 1, a_n^t = ch_m, \left|a_n^t \cup a_{\bar{n}}^t\right| = c_m^t, \\ -1, s_m^t = -1. \end{cases} \tag{4}$$

where the function $f((c_m^t)/(N))$ is defined in the case when the channel available is chosen. With regard to the ratio of the number of agents allocated on the same channel ch_m, $m \in [1, M]$ to the total number of agents, $((c_m^t)/(N))$, it guides the agent to access idle channel properly. And $\bar{n}$ denotes the agents except for agent n, $|\cdot|$ measures the number of agents on ch_m. On the contrary, when a busy channel is wrongly chosen by agent n, a negative reward -1 is occurred and an error identification is informed.

Specifically, $f(\cdot)$ denotes a piecewise function bounded on $1/K$, which is set to guarantee that all the available channels can be utilized fairly. The function is formulated as

$$f\left(\frac{c_m^t}{N}\right) = \begin{cases} (K-1)\frac{c_m^t}{N}, 0 < \frac{c_m^t}{N} < \frac{1}{K}, \\ \frac{1}{K}\left(\frac{N}{c_m^t} - 1\right), \frac{1}{K} < \frac{c_m^t}{N} < 1. \end{cases} \tag{5}$$

When $((c_m^t)/(N))$ increases, the reward increases at first and achieves the maximum value at the boundary $1/K$, then decreases rapidly. It signifies that the agent n will get a small reward whether there are too many or too few agents on the selected ch_m. Whereas a balance scheme is explored for all agents in the cognitive network under the limited channels.

Based on the immediate rewards from all agents, the total reward in one slot can be written as

$$r_{\text{tot}}^t = \sum_{n=1}^{N} r_n^t(\tau_n, a_n), \tag{6}$$

where τ_n denotes the observation-action history of agent n. Actually, each agent in the network is supposed to action toward the whole optimal DSA. We call it a cooperative game, which is a special type of exact potential game

(EPG). Based on this theory, the monotonicity of r_{tot}^t conforms to that of r_n^t [40]. Thus we have

$$\underset{\boldsymbol{a}}{\operatorname{argmax}} r_{\text{tot}}^t(S^t, \boldsymbol{a}) = \begin{pmatrix} \underset{a_1}{\operatorname{argmax}} r_1^t(\tau_1, a_1) \\ \underset{a_2}{\operatorname{argmax}} r_2^t(\tau_2, a_2) \\ \vdots \\ \underset{a_N}{\operatorname{argmax}} r_N^t(\tau_N, a_N) \end{pmatrix}. \tag{7}$$

In the finite time slots, the global reward of all agents can be obtained as

$$R_{\text{tot}} = \sum_{t=1}^{T} \sum_{n=1}^{N} \gamma^{t-1} r_n^t(\tau_n, a_n), \tag{8}$$

where γ is the discount factor, reflecting the influence of the agents' action at the current time slot on the long-term return. And equation (8) can be simply transformed into

$$R_{\text{tot}} = \sum_{t=1}^{T} \gamma^{t-1} \sum_{n=1}^{N} r_n^t(\tau_n, a_n), \tag{9}$$

which can be furtherly integrated as

$$R_{\text{tot}} = \sum_{t=1}^{T} \gamma^{t-1} r_{\text{tot}}^t. \tag{10}$$

The ultimate goal of each agent is to obtain their own optimal spectrum sense and DSA strategies $\pi^* \triangleq (\pi_1^*, \cdots, \pi_N^*)$, so as to maximize the expected cumulative reward of the whole network. The corresponding problem can be formulated as

$$\text{P1}: \pi^* = \underset{\pi:\{a_1,\cdots,a_N\}}{\operatorname{argmax}} E(R_{\text{tot}}), \tag{11}$$

where $E(\cdot)$ denotes the expectation.

From the above defined problem, the agents are expected to possess their excellent abilities of independent perception and decision to maximize a global cumulative reward in a cooperative manner. It is challenging since there is not even a central node to control the whole allocation in practical scenario and no direct information exchange beforehand among agents.

4. QMIX Algorithm for DSA

4.1. Algorithm Description. We consider the QMIX algorithm [41] with the CTDE structure to solve the DSA problem. There are two phases for the DSA of all the agents, i.e., offline training and online execution, respectively. For one thing, in the offline phase, agents perceive environmental information and offload the DSA task to the MEC server, who is responsible to train and issue the distributed DSA strategies by computing and analyzing the received data. For another, in the online phase, the MEC server keeps silence, and each agent executes action autonomously by the learnt strategy.

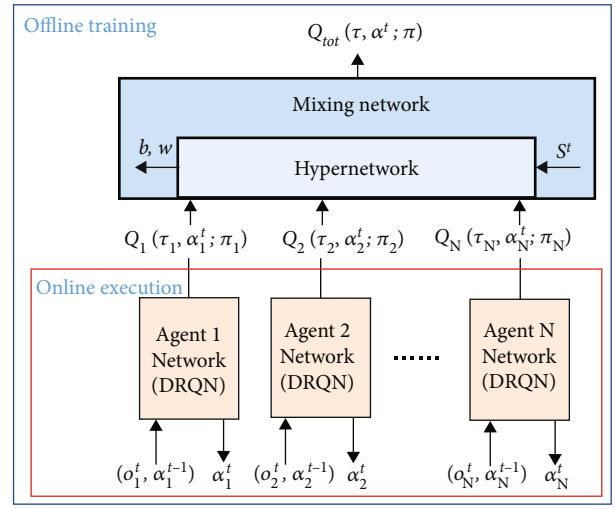

Figure 3: The structure of QMIX algorithm.

As shown in Figure 3, there are N local agent networks for SUs and one mixing network deployed at the central controller. And the agent networks are constructed by deep recurrent Q-network (DRQN), catering for the agents' partial observation. Here, as the agents considered in the network are homogeneous and also for the system stability, all the DRQNs are equipped with the same network structure and parameters. For any agent $n \in [1, N]$, with the current observation o_n^t and the previous action a_n^{t-1}, the local action value function Q_n is obtained, which enables the agent to choose action a_n^t. Then all the agents' value functions $\{Q_n(\tau_n, a_n^t; \pi_n) | n = 1, 2, \cdots, N\}$ are injected into the mixing network. Note that a hypernetwork is embedded in the mixing network, which makes full use of the global channel state S^t to improve the convergence speed and output the parameters, e.g., bias and nonnegative weight (b, w) for the mixing network. Finally, by the nonlinear map model of the mixing network, the joint action value function $Q_{\text{tot}}(\tau, \boldsymbol{a}^t; \pi)$ is produced.

The advantage of this method is that the monotonicity of Q_{tot} and Q_n can remain the same, i.e.,

$$\frac{\partial Q_{\text{tot}}}{\partial Q_n} \geq 0, \forall n \in [1, N], \tag{12}$$

which well coincides with the property of the problem. Therefore, the relationship between Q_{tot} and Q_n can be furtherly written as

$$\underset{a}{\operatorname{argmax}} Q_{\text{tot}}(\boldsymbol{\tau}, \boldsymbol{a}^t; \boldsymbol{\pi}) = \begin{pmatrix} \underset{a_1^t}{\operatorname{argmax}} Q_1(\tau_1, a_1^t; \pi_1) \\ \underset{a_2^t}{\operatorname{argmax}} Q_2(\tau_2, a_2^t; \pi_2) \\ \vdots \\ \underset{a_N^t}{\operatorname{argmax}} Q_N(\tau_N, a_N^t; \pi_N) \end{pmatrix}. \tag{13}$$

1: **Initialization:**
The network environment and experience replay buffer $\mathscr{R}$; the parameters for hypernetwork and all of the agent networks π;
2: **Setting:**
The target-network parameters $\bar{\pi} = \pi$, the learning rate α, the discount factor γ, the batch size l, maximum training epoch, episode, slot: Epo, Epi, T, maximum train step L;
3: [**Centralized Training Phase**]:
4: **while** *epoch* ≤ E*po***do**
5: **for** *episode* = 1, ⋯, Epi**do**
6: **for** $t = 1, \cdots,$ T**do**
7: **for** each agent n**do**
8: Get observation o_n^t, action a_n^t, reward r_n^t;
9: **end for**
10: Get the next observation o^{t+1};
11: Store the o^t, a^t, r^t, o^{t+1} to the observation-action history;
12: **end for**
13: Store the episode data to the replay buffer $\mathscr{R}$;
14: **end for**
15: **for** *train* ≤ L in each epoch **do**
16: Sample a batch of l episodes' experience from $\mathscr{R}$;
17: **for** each slot in each sampled episode **do**
18: Get Q_t and Q_{t+1} from the evaluate-network and the target-network, respectively;
19: **end for**
20: Calculate the loss function by (14), and update the evaluate-network parameters $\pi = \pi - \alpha \nabla_\pi L(\pi)$;
21: Update the target-network parameters $\bar{\pi} = \pi$;
22: **end for**
23: Save DRQN and QMIX network models;
24: **end while**
25: [**Decentralized Executing Phase**]:
26: **Setting**: *loadmodel* = TRUE;
27: **Input**: The channel state;
28: **Output**: The agents' observations and actions.

Algorithm 1: The proposed QMIX-based DSA algorithm.

By learning the optimal joint value function Q_{tot}, we can obtain the agents' local distributed strategies indirectly. And the update criteria for Q_{tot} is to minimize the loss function $L(\pi)$, which is given by

$$L(\pi) = \sum_{i=1}^{l} \left[y_{\text{tot}}^i - Q_{\text{tot}}\left(\tau, \boldsymbol{a}, S^t; \pi\right) \right]^2, \tag{14}$$

where y_{tot}^i is expressed as

$$y_{\text{tot}}^i = R_{tot}^i + \gamma \max_{\boldsymbol{a}'} \bar{Q}\left(\tau', \boldsymbol{a}', S^{t\prime}; \bar{\pi}\right), \tag{15}$$

where $\bar{Q}(\tau', \boldsymbol{a}', S^{t\prime}; \bar{\pi})$ denotes the target network, supplying for the stable training.

Specifically, the process of the proposed QMIX-based DSA algorithm is listed as Algorithm 1.

4.2. Computational Complexity Analysis. In the proposed QMIX-based DSA algorithm, DRQN is adopted for each agent, which can well handle the Dec-POMDP problem. Besides, some simple activation function, e.g., ReLU and ELU are employed in the algorithm. The operation mainly involves matrix multiplication and addition. In particular, for the DRQN, let us assume that there are V layers, and the number of neural units is $u_v (v \in [1, V])$ in v-th layer, and Z is the size of input layer. Then, the number of multiplications through DRQN can be presented as $W = Z \cdot v_1 + \sum_{v=1}^{V-1} u_v \cdot u_{v+1}$. For each agent, the computational complexity of one sample is $O(W)$. Note that the offline training is

Table 2: Simulation parameters.

Parameters	Values
Number of PUs	1
Number of SUs	9
Number of MEC servers	1
Number of authorized channels	4
Discount factor γ	0.99
Learning rate α	$5 * 10^{-4}$
Replay buffer capacity	100 episodes of experience data
Batch size l	16 episodes of experience data
Updating step	40 steps
Training epoch Epo	2000 epochs
Training episode Epi	100 episodes
Training slot T	20 slots
Exploration probability ε	0.4 to 0.02

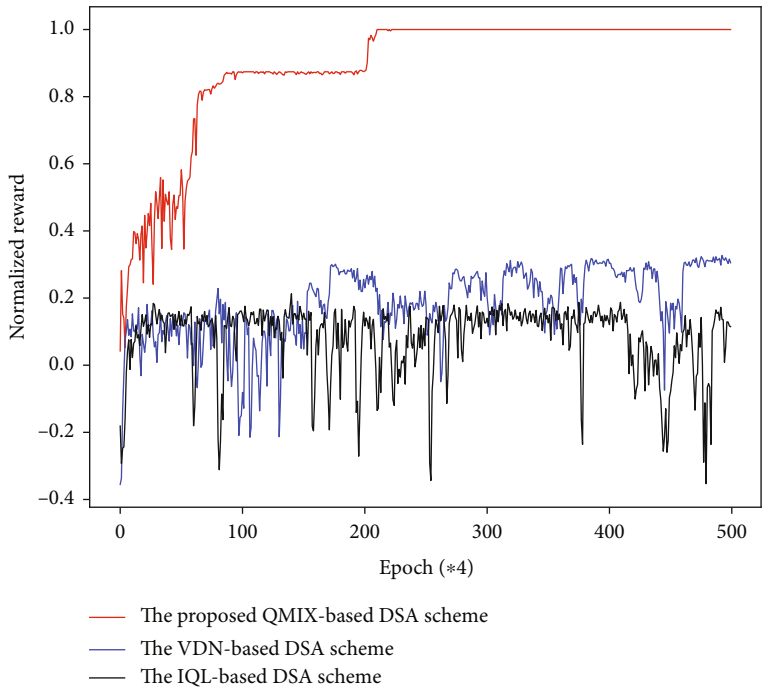

Figure 4: Normalized reward versus training epoch under different schemes.

parallelly worked at the edge server in the training phase. The training complexity of one batch of l episodes under T training slots is $O(lTW)$. And the whole computational complexity is $O(IlTW)$ until the algorithm converges over I iterations. Further, the computational complexity of the execution phase is $O(W)$, since each SU acts locally at each time step. Due to its monotonicity, the complexity increases linearly with the increase of input scale, which greatly improves the efficiency of the algorithm. Therefore, less computational resource is required in practice.

5. Simulation Results

In this section, we provide the simulation parameter setting, and then evaluate the performance of the proposed QMIX-based DSA scheme and the rationality of the defined problem via simulation.

Since there is a mixing network, a hypernetwork and N agent networks in the proposed algorithm, the corresponding network parameters setting are illustrated as follows: the mixing network, which brings the global action value from local action values, has one hidden layer of 32 neurons, and the nonlinear function ELU is employed as the activation function. For the hypernetwork, it consists of one hidden layer which has 64 neurons with ReLU as the activation function. Each agent network is with one recurrent layer employing a GRU with 64-dimension hidden state. Unless otherwise specified, other simulation parameters are summarized in Table 2.

In particularly, for hyperparameters, the replay buffer is capacity-limited which can store 100 sets of data, and the oldest data will be removed when the buffer is full. The batch size l for sampling is 16 episodes. The target networks of the DRQN are updated every 40 training steps. In the whole process, it is considered that there are 2000 training epochs, each epoch has 100 episodes, in which 20 time slots are regarded as one episode. In the training phase, to encourage the agent to explore the environment, the "explore and exploit" mechanism is employed by agents to choose actions [42]. The exploration probability ε decays from 0.4 to 0.02 over 400 steps. Then, in order to timely evaluate the quality of the training performance, the distributed execution is conducted every four training epochs, where we set $\varepsilon = 0$, and the agents make decisions only by the local models. For the environment setting, we assume that the channels change in periodic mode and each SU can only sense one channel.

To verify the advantages of the proposed scheme, in Figure 4, we compared our proposed scheme with two other schemes: (1) the IQL-based DSA scheme [43] and (2) the VDN-based DSA scheme [44]. We take nine SUs and four channels with one channel unavailable to compare the performance of these schemes. The abscissa is the training epoch, and the ordinate is the normalized reward. The simulation results show that, the performance curves of IQL and VDN based schemes show relatively large fluctuation, which are far from the best effect. It can be seen that the maximum value under IQL is only 0.17, while VDN is better than IQL reaching just 0.32. The reasons for this result can be explained as follows: for IQL-based scheme, each agent operates independently in the whole learning process, which is not conducive to the stability and convergence. And for VDN-based scheme, it does not use global state information during central training, and a simple weighted summation method is used to decompose the joint value function to update the agents' strategies, causing a bad training effect. In addition, due to the powerful fitting ability and the integration of global environment information, an excellent DSA effect is achieved in our proposed QMIX-based scheme. Therefore, the DSA performance of our proposed QMIX-based DSA scheme outperform other two schemes.

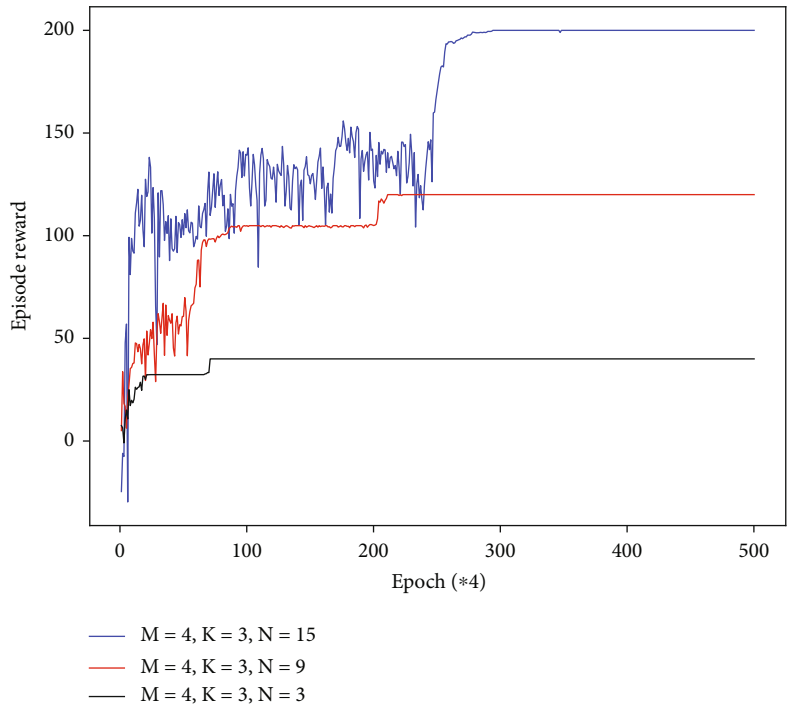

Figure 5: Episode reward versus training epoch under different number of SUs.

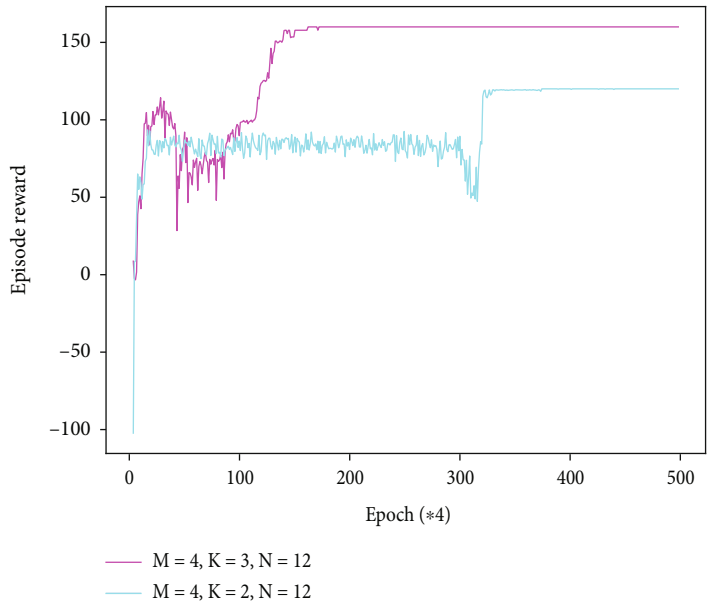

Figure 6: Episode reward versus training epoch under different channels available.

Figure 5 displays the sum of rewards obtained by all users in an episode versus training epoch in the case of different number of SUs. There are four channels considered in the network, where there are always three channels available for SUs that changes periodically over time. It can be seen intuitively that under three different settings, as the number of training epoch increases, the total episode reward increases gradually, and finally reaches to the maximum value within limited training. Note that the negative reward happens at the beginning of the training, which can be explained that some SUs select the nonidle channels, since the agents' network models are still rough at that moment, although four epochs' training is done. Specially, when there are 15 SUs, the initial episode reward is about -30, and then a longest time is experienced for convergence. This is because the more number of SUs means the larger calculation dimension and the slower learning speed, which makes it more difficult for SUs to learn and analyze the environment. That is, through the proposed method, firstly, the SUs have learned to capture the spectrum holes. On this basis, the load balance is realized, so as to obtain the

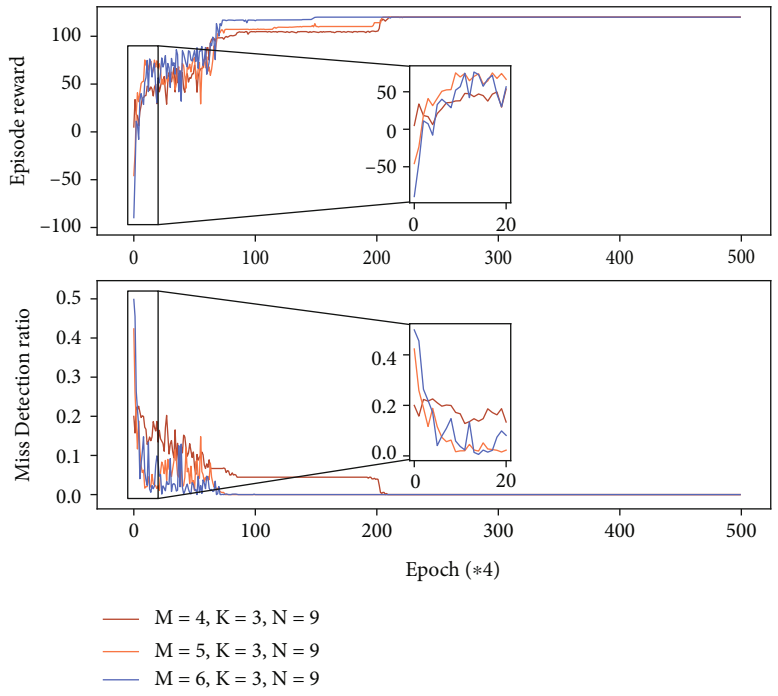

Figure 7: Episode reward and miss detection ratio versus training epoch under different number of authorized channels.

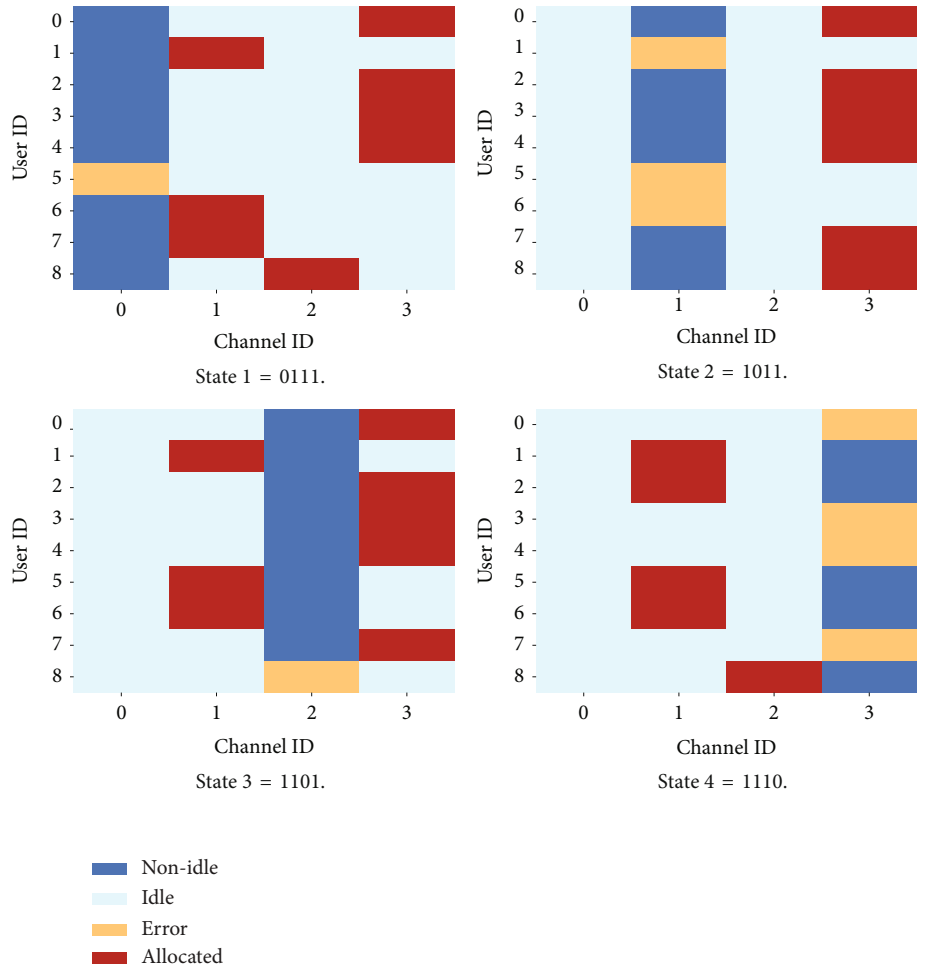

Figure 8: The distributed execution after the initial training.

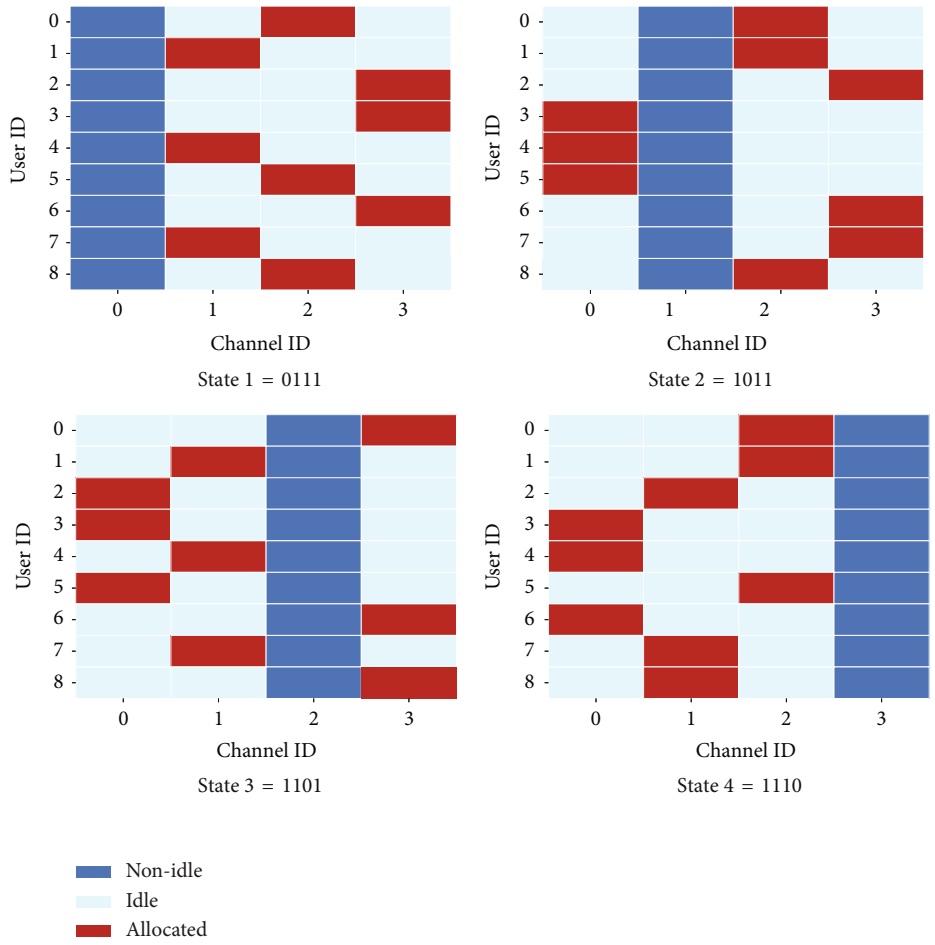

Figure 9: The distributed execution after the final training.

optimal DSA. Besides, we can observe that, when the networks are well trained, more episode reward value of the system can be achieved with more users. It is related to the definition of global reward in cooperative MARL environment, which integrates all SUs' rewards.

In Figure 6, the behavior of the episode rewards under different number of idle channels is plotted. To facilitate the comparison, we set the idle channels $K = 2, 3$ for $M = 4$, and 12 SUs are participating in the DSA. Likewise, we can observe that the curves fluctuate but overall increase and then converge to the maximum value as the training epoch increases. As for the situation of the fewer channels available, it can be seen that a slower convergence speed is acquired, which means the fewer optional channels requires more time for multiple SUs to make a "trial and error" until the model is well-done, so as to achieve the optimal DSA. We can also observe that at the beginning of the training, a quite low negative value, about -100, is caused when $K = 2$, i.e., there are two unavailable channels. It can be explained that the more number of unavailable channels, the greater probability that the SUs make a wrong decision to choose unavailable channels. It also reflects the huge influence of the number of unavailable channels on network learning effect. What is more, it is also shown that the more available channels, the larger total episode reward is obtained, which reveals that the more available channels brings the better balance among these channels.

Figure 7 evaluates the performance with respect to number of authorized channels, which are set as four, five, and six, and the PU occupies one, two, or three channels correspondingly to ensure the same channels available for SUs. Considering that the SUs' sense accuracy is also an important indicator for the DSA, the miss detection ratio is evaluated as well. From the presented reward curves, we can observe that in three different cases, the episode rewards are increasing gradually. Since there is little difference among the set of available channels and SUs, the curves are entangled with each other, and the values are relatively close in the whole process. Finally, they all converge at the same value, which is also the optimal value after the network model is fully trained.

Meanwhile, the miss detection ratio behaves an opposite trend. This intuitively shows that the agents' detection ability is indeed weak as they experience little learning, which makes it easy for them to make the error decision. And we find that, when more channels are occupied by the PU, the initial miss detection ratio is greater, which is consistent with the reward at the starting stage. Then, with the increase of training time, the SUs achieve a perfect detection (miss detection ratio = 0). At this time, compared with reward curves, it can be seen that the reward value has not reached the optimal value, but

converges after more training. This shows that after overcoming the imperfect detection, it takes some time to further realize load balance for SUs on the available channels.

To be more intuitive, Figures 8 and 9 present the effects of the distributed execution in four consecutive channel states under the initial and final training phases, respectively. Here, the initial phase is the first four training epochs, while the final phase is the last epoch when the models are fully trained. We assume that nine SUs independently make a choice from four authorized channels in different channel states for instance.

We can see from Figure 8 that there are many wrong channel detections and selections in the fourth tested states. Due to the fact that the SUs have no more knowledge of the channel variation characteristics in the initial training phase, they access channels almost randomly. Specially, in the state2, although there are three idle channels, the SUs gather in the fourth channel. This reduces the channel utilization and also causes congestion of other idle channels. Whereas, we can see from Figure 9 that the result of DSA is well done after the final training. Not only no one selects the occupied channel but also the SUs on each available channels are balanced. The goal of the DSA in the considered scenario is achieved, and the effectiveness of proposed algorithm is demonstrated. Besides, we have counted the real execution time under one of the channel states, and we find that it takes only about 26.59 ms for all SUs' DSA. It highlights the low computational complexity of the proposed algorithm furtherly. Then, since all SUs' collected spectrum data has no error after training, the availability, accuracy, and timeliness of the acquired spectrum data can be guaranteed.

6. Conclusion

In this paper, we have studied the distributed DSA strategies for multiple cognitive users in MEC-enabled network, where the spectrum environment is time-varying, and the users make decisions with imperfect spectrum sensing. The DSA task, including capturing the spectrum hole and achieving the load balance on channels available, is investigated. We modeled the problem as Dec-POMDP, and a QMIX-based DSA algorithm is proposed, which allows users offload their task to the MEC server to train the network models. We evaluated the system reward and the miss detection ratio of the DSA by the proposed algorithm. The results showed the rationality of the model and the effectiveness of the proposed algorithm.

Data Availability

The data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Acknowledgments

This work was supported by National Natural Science Foundation of China under Grant Nos. 62171449, 61931020 and 62001483.

References

[1] P. Mach and Z. Becvar, "Mobile edge computing: a survey on architecture and computation offloading," *IEEE Communications Surveys Tutorials*, vol. 19, no. 3, pp. 1628–1656, 2017.

[2] U. Drolia, R. Martins, J. Tan et al., "The case for mobile edge-clouds," in *2013 IEEE 10th International Conference on Ubiquitous Intelligence and Computing and 2013 IEEE 10th International Conference on Autonomic and Trusted Computing*, pp. 209–215, Vietri sul Mare, Italy, 2013.

[3] Z. Ning, Y. Yang, X. Wang et al., "Dynamic computation offloading and server deployment for UAV-enabled multi-access edge computing," *IEEE Transactions on Mobile Computing*, p. 1, 2021.

[4] X. Tang, Z. Wen, and J. E. A. Chen, "Joint optimization task offloading strategy for mobile edge computing," in *2021 IEEE 2nd international conference on information technology, Big Data and Artificial Intelligence (ICIBA)*, pp. 515–518, Chongqing, China, 2021.

[5] X. Wang, Z. Ning, S. Guo, M. Wen, L. Guo, and V. Poor, "Dynamic UAV deployment for differentiated services: a multi-agent imitation learning based approach," *IEEE Transactions on Mobile Computing*, p. 1, 2021.

[6] H. Zhao, K. Ding, N. I. Sarkar, J. Wei, and J. Xiong, "A simple distributed channel allocation algorithm for D2D communication pairs," *IEEE Transactions on Vehicular Technology*, vol. 67, no. 11, pp. 10960–10969, 2018.

[7] J. Li, H. Zhao, J. Wei, D. Ma, and L. Zhou, "Sender-jump receiver-wait: a simple blind rendezvous algorithm for distributed cognitive radio networks," *IEEE Transactions on Mobile Computing*, vol. 17, no. 1, pp. 183–196, 2018.

[8] J. Li, H. Zhao, S. Zhang, A. S. Hafid, D. Niyato, and J. Wei, "Cross-Layer analysis and optimization on access delay in channel-hopping-based distributed cognitive radio networks," *IEEE Transactions on Communications*, vol. 67, no. 7, pp. 4654–4668, 2019.

[9] S. Zhang, A. S. Hafid, H. Zhao, and S. Wang, "Cross-layer rethink on sensing-throughput tradeoff for multi-channel cognitive radio networks," *IEEE Transactions on Wireless Communications*, vol. 15, no. 10, pp. 6883–6897, 2016.

[10] W. Zhang, Y. Sun, L. Deng, C. K. Yeo, and L. Yang, "Dynamic spectrum allocation for heterogeneous cognitive radio networks with multiple channels," *IEEE Systems Journal*, vol. 13, no. 1, pp. 53–64, 2019.

[11] T. H. Lim, W. S. Jeon, and D. G. Jeong, "Centralized channel allocation scheme in densely deployed 802.11 wireless LANs," in *2016 18th International Conference on Advanced Communication Technology (ICACT)*, pp. 249–253, PyeongChang, Korea, 2016.

[12] P. Shetkar and S. B. Ronghe, "Spectrum sensing and dynamic spectrum allocation for cognitive radio network," in *2018 4th International Conference for Convergence in Technology (I2CT)*, pp. 1–5, Mangalore, India, 2018.

[13] Z. Ning, S. Sun, X. Wang et al., "Blockchain-enabled intelligent transportation systems: a distributed crowdsensing framework," *IEEE Transactions on Mobile Computing*, 2021.

[14] Q. Cong and W. Lang, "Deep multi-user reinforcement learning for centralized dynamic multichannel access," in *2021 6th International Conference on Intelligent Computing and Signal Processing (ICSP)*, pp. 824–827, Xi'an, China, 2021.

[15] Q. Cong and W. Lang, "Double deep recurrent reinforcement learning for centralized dynamic multichannel access," *Wireless Communications and Mobile Computing*, vol. 2021, Article ID 5577756, 10 pages, 2021.

[16] J. Li, H. Zhao, A. S. Hafid, J. Wei, H. Yin, and B. Ren, "A bio-inspired solution to cluster-based distributed spectrum allocation in high-density cognitive internet of things," *IEEE Internet of Things Journal*, vol. 6, no. 6, pp. 9294–9307, 2019.

[17] X. Pei, "A novel dynamic spectrum allocation and power control algorithm with fairness for multi-hop cognitive radio networks," in *2015 IEEE 5th International Conference on Electronics Information and Emergency Communication*, pp. 440–443, Beijing, China, 2015.

[18] M. M. A. Osman, S. K. S. Yusof, and N. N. N. Abd Malik, "Load balanced clustering algorithm for cognitive radio ad hoc networks," in *2018 2nd International Conference on Telematics and Future Generation Networks (TAFGEN)*, pp. 43–48, Kuching, Malaysia, 2018.

[19] P. Li, Z. Zhao, D. Liu, and D. Hou, "The research of dynamic spectrum allocation based on game theory," in *2018 IEEE 3rd advanced information technology, Electronic and Automation Control Conference (IAEAC)*, pp. 14–17, Chongqing, China, 2018.

[20] P. Li, B. Han, H. Li, D. Hou, D. Liu, and G. Wang, "The research of dynamic spectrum allocation based on Nash bargaining game," in *2018 IEEE 4th Information Technology and Mechatronics Engineering Conference (ITOEC)*, pp. 70–74, Chongqing, China, 2018.

[21] E. E. Tsiropoulou, A. Kapoukakis, and S. Papavassiliou, "Energy efficient subcarrier allocation in SC-FDMA wireless networks based on multilateral model of bargaining," in *2013 IFIP Networking Conference*, pp. 1–9, Brooklyn, NY, USA, 2013.

[22] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, The MIT Press, 2nd edition, 2018.

[23] M. Sohaib, J. Jeong, and S.-W. Jeon, "Dynamic multichannel access via multi-agent reinforcement learning: throughput and fairness guarantees," *ICC 2021 - IEEE International Conference on Communications*, pp. 1–6, 2021.

[24] J. Chen, Z. Gao, and Y. Xu, "Opportunistic spectrum access with limited feedback in unknown dynamic environment: a multi-agent learning approach," in *The 2014 5th International Conference on Game Theory for Networks*, pp. 1–6, Beijing, China, 2014.

[25] 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 of Things Journal*, vol. 6, no. 2, pp. 1938–1948, 2019.

[26] H. Li, "Multi-agent q-learning for competitive spectrum access in cognitive radio systems," in *2010 Fifth IEEE Workshop on Networking Technologies for Software Defined Radio Networks (SDR)*, pp. 1–6, Boston, MA, USA, 2010.

[27] H. Agrawal and K. Asawa, "Decentralized learning for opportunistic spectrum access: multiuser restless multiarmed bandit formulation," *IEEE Systems Journal*, vol. 14, no. 2, pp. 2485–2496, 2020.

[28] Y. Li, Y. Wu, and W. Jia, "Dynamic spectrum allocation enabled multiuser latency minimization in mobile edge computing," in *2020 16th International Conference on Mobility, Sensing and Networking (MSN)*, pp. 161–168, Tokyo, Japan, 2020.

[29] Q. Li, Y. Sun, Z. Hao, and Y. Zhang, "Energy efficient spectrum resource allocation in mobile edge computing," in *2019 IEEE 5th International Conference on Computer and Communications (ICCC)*, pp. 1114–1119, Chengdu, China, 2019.

[30] Z. Ning, S. Sun, X. Wang et al., "Intelligent resource allocation in mobile blockchain for privacy and security transactions: a deep reinforcement learning based approach," *Science China Information Sciences*, vol. 64, no. 6, pp. 1–6, 2021.

[31] U. Kaytaz, S. Ucar, B. Akgun, and S. Coleri, "Distributed deep reinforcement learning with wideband sensing for dynamic spectrum access," in *2020 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–6, Seoul, Korea, 2020.

[32] O. Naparstek and K. Cohen, "Deep multi-user reinforcement learning for dynamic spectrum access in multichannel wireless networks," in *GLOBECOM 2017 - 2017 IEEE Global Communications Conference*, pp. 1–7, Singapore, 2017.

[33] X. Tan, L. Zhou, H. Wang et al., "Cooperative multi-agent reinforcement learning based distributed dynamic spectrum access in cognitive radio networks," *IEEE Internet of Things Journal*, 2022.

[34] X. Wang, Z. Ning, S. Guo, M. Wen, and H. V. Poor, "Minimizing the age-of-critical-information: an imitation learning-based scheduling approach under partial observations," *IEEE Transactions on Mobile Computing*, vol. 21, no. 9, pp. 3225–3238, 2022.

[35] O. H. Toma, M. Lpez-Bentez, D. K. Patel, and K. Umebayashi, "Reconstruction algorithm for primary channel statistics estimation under imperfect spec trum sensing," in *2020 IEEE Wireless Communications and Networking Conference (WCNC)*, pp. 1–5, Seoul, Korea, 2020.

[36] D. Silver and J. Veness, "Monte-Carlo planning in large POMDPs," in *Advances in Neural Information Processing Systems 23: 24th Annual Conference on Neural Information Processing Systems*, pp. 2164–2172, Vancouver, British Columbia, Canada, 2010.

[37] A. Somani, N. Ye, D. Hsu, and W. S. Lee, "DESPOT: online POMDP planning with regularization," *Advances in Neural Information Processing Systems*, vol. 58, 2013.

[38] P. Cai, Y. Luo, D. Hsu, and W. S. Lee, "Hyp-DESPOT: a hybrid parallel algorithm for online planning under uncertainty," *Robotics: Science and Systems Foundation*, vol. 40, no. 2-3, pp. 558–573, 2021.

[39] M. Gupta and G. Yerma, "Improved weighted cooperative spectrum sensing algorithm based on reliability in cognitive radio networks," in *IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology*, pp. 609–612, Bangalore, India, 2017.

[40] D. Monderer and L. S. Shapley, "Potential games," *Games and Economic Behavior*, vol. 14, no. 1, pp. 124–143, 1996.

[41] T. Rashid, M. Samvelyan, C. Schroeder, G. Farquhar, J. Foerster, and S. Whiteson, *Qmix: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning*, PMLR, 2018.

[42] E. R. Gomes and R. Kowalczyk, "Modelling the dynamics of multiagent Q-learning with greedy exploration," in

Proceedings of the International Conference on Autonomous Agents & Multiagent Systems, pp. 1181-1182, Budapest, Hungary, 2009.

[43] M. Tan, "Multi-agent reinforcement learning: Independent vs. cooperative agents," in *The 10th International Conference on Machine Learning*, pp. 330–337, Amherst: University of Massachusetts, 1993.

[44] P. Sunehag, G. Lever, A. Gruslys et al., "Value-decomposition networks for cooperative multi-agent learning based on team reward," in *The 17th International Conference on Autonomous Agents and Multiagent Systems*, pp. 2085–2087, Stockholm, Sweden, 2017.