IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 22, 2020, accepted February 5, 2020, date of publication February 11, 2020, date of current version February 24, 2020.

*Digital Object Identifier 10.1109/ACCESS.2020.2973140*

# Deep Reinforcement Learning-Based Channel Allocation for Wireless LANs With Graph Convolutional Networks

**KOTA NAKASHIMA, (Student Member, IEEE), SHOTARO KAMIYA, (Student Member, IEEE), KAZUKI OHTSU, (Student Member, IEEE), KOJI YAMAMOTO, (Member, IEEE), TAKAYUKI NISHIO, (Member, IEEE), AND MASAHIRO MORIKURA, (Member, IEEE)**
Graduate School of Informatics, Kyoto University, Kyoto 606-8501, Japan

Corresponding author: Koji Yamamoto (kyamamot@i.kyoto-u.ac.jp)

This work was supported in part by the Japan Society for the Promotion of Science (JSPS) KAKENHI under Grant JP18H01442.

**ABSTRACT** For densely deployed wireless local area networks (WLANs), this paper proposes a deep reinforcement learning-based channel allocation scheme that enables the efficient use of experience. The central idea is that an objective function is modeled relative to communication quality as a parametric function of a pair of observed topologies and channels. This is because communication quality in WLANs is significantly influenced by the carrier sensing relationship between access points. The features of the proposed scheme can be summarized by two points. First, we adopt graph convolutional layers in the model to extract the features of the channel vectors with topology information, which is the adjacency matrix of the graph dependent on the carrier sensing relationships. Second, we filter experiences to reduce the duplication of data for learning, which can often adversely influence the generalization performance. Because fixed experiences tend to be repeatedly observed in WLAN channel allocation problems, the duplication of experiences must be avoided. The simulation results demonstrate that the proposed method enables the allocation of channels in densely deployed WLANs such that the system throughput increases. Moreover, improved channel allocation, compared to other existing methods, is achieved in terms of the system throughput. Furthermore, compared to the immediate reward maximization method, the proposed method successfully achieves greater reward channel allocation or realizes the optimal channel allocation while reducing the number of changes.

**INDEX TERMS** Wireless LAN, channel allocation, deep reinforcement learning, graph convolutional networks, replay buffer.

## I. INTRODUCTION

Channel allocation is an important problem in densely deployed wireless local area networks (WLANs) owing to the large number of access points (APs) and limited available channels. A poor channel allocation causes substantial contention among the APs and stations (STAs), and reduces the throughput of each AP. WLAN channel allocation schemes in centrally managed environments have been proposed [1]. IEEE 802.11 Task Group be (TGbe) focuses on a multiple increase in real-time applications that impose strict requirements on packet transmission delay and packet loss ratio [2]. To meet the requirements, new resource allocation algorithms are required. Coordinated AP control methods in densely deployed WLANs have been discussed. The poor channel allocation issue can be avoided by effective channel allocation with a limited number of channels. This is the motivation for this study.

The associate editor coordinating the review of this manuscript and approving it for publication was Ayaz Ahmad.

**TABLE 1. Coordinated channel allocation approaches for WLANs. Existing approaches do not focus on improving the cumulative throughput.**

| | Immediate | Cumulative |
|---|---|---|
| Optimization-based | DSATUR [3] | × |
| Observation-based | MBLC [4] | **Reinforcement learning** |

Coordinated channel allocation approaches have been proposed, including the DSATUR [3] and Measurement-Based Local-Coord (MBLC) [4] methods as indicated in Table 1.

DSATUR is a graph coloring approach, where nodes and edges represent APs and the contentions among APs (i.e., carrier sensing relationship), respectively, and the color of each node represents its channel. Note that this approach indirectly improves the throughput by reducing the number of adjacent APs using the same channel.

To directly improve throughput, observations of the throughput are required because it is difficult to model the throughput as an explicit function of channels in general. MBLC [4], an immediate throughput improving approach based on observations, uses a weighted cost function of the observed interference based on the physically measured interference power on all channels. This approach switches the channel of an AP exclusively if the calculated weighted interference does not increase after the operation. The objective of this approach is to maximize the immediate throughput at the time of changing the channel of an AP. However, the throughput is not necessarily maximized at the end of the sequence because they can fall into a local optimal allocation.

Following, we introduce prior studies that addressed channel allocation problems in WLANs. In [5], a channel assignment scheme was proposed to maximize the signal-to-interference ratio (SIR) at the user level. This scheme focused on the load balancing of APs according to the number of users of each AP. An optimal channel allocation algorithm in dynamic channel bonding WLANs was proposed in [6]. This algorithm achieved an improvement in the throughput by operating the bandwidth of each channel without overlapping. Raschellà *et al.* [7] evaluated a channel assignment algorithm. The objective was to minimize the parameters that represented the interference among the APs. As a distributed channel allocation method, potential game-based [8] methods are summarized in [9]. In these approaches, to achieve the Nash equilibrium, each AP stochastically selects its channel. As a potential game-based method for WLANs, Xu *et al.* [10] proposed an approach to minimize the carrier sensing relationships among the APs. Suliman *et al.* [11] indicated the potential of artificial intelligence in solving channel allocation problems in wireless communications. This study addressed the determination of the minimum number of channels to satisfy the demands of a network using an artificial immune system. Ghahfarokhi [12] introduced a distributed channel assignment algorithm using a machine learning algorithm, learning automata [13]. The objective of this study was to improve the quality of the experience and user-level fairness. Jeunen *et al.* [14] proposed a machine learning approach using a combination of airtime overlap minimization and bad neighbor detection, which identified devices interfering with each other. This method also focused on improving the experience for users.

As far as we know, the research problem of this paper, which is the direct maximization of the system throughput in WLANs based on the observed throughput, has not been studied before. This is because the prior studies aimed to improve the throughput indirectly (e.g., by reducing the number of carrier sensing relationships or by improving the SIR). Because the throughput of WLANs is influenced by various factors, these prior studies did not essentially allocate channels to maximize the throughput.

Reinforcement learning is a possible solution to allocate channels based on the feedback of the measured throughput considering the allocation sequence. Reinforcement learning is a decision-making process that learns what choice provides greater reward based on the experience. In particular, deep reinforcement learning has attracted considerable attention and has been utilized to achieve channel allocation in wireless networks [15] because of the effectiveness of the function approximation. In [16], a deep reinforcement learning-based channel allocation method was proposed. However, this method addressed operating only one AP, i.e., this method did not consider a centralized channel allocation problem; rather, it addressed a distributed problem. According to [15], several of the deep reinforcement learning-based channel allocation studies have considered distributed channel allocation or the dynamic spectrum access problem. There would appear to be no previous study for the coordinated WLAN channel allocation problem.

This paper proposes a deep reinforcement learning-based scheme that is suitable for coordinated channel allocation problems in densely deployed WLANs. In reinforcement learning, states must be adequately associated to rewards because the agent acts based on the observed states; thus, we must carefully design the states. To improve the throughput in WLAN channel allocation problems, it is important to capture the carrier sensing relationships among the APs, as in the graph coloring approach of DSATUR. Therefore, we design the states based on the carrier sensing relationships among APs and used channels to allow an agent to associate a given reward relative to throughputs with the designed states. Furthermore, we require function approximation to manage the enormous number of observable states in densely deployed WLANs. In this context, extracting input features is important for the improvement of the learning performance. For example, convolutional neural network (CNN) is an effective method to improve the performance of a neural network, which is commonly used to extract the features of input images, (e.g., AlphaGo [17]). We demonstrate that the function approximation based on a simple neural network does not provide sufficient performance for the channel allocation problem in Section VI. To extract the features of the adjacent APs and used channels, we propose incorporating graph convolutional layers [18]–[21] into a neural network. Because the states based on carrier sensing relationships can be considered as a graph signal, a graph convolutional network (GCN) is suitable for our settings. Note that CNN is not suitable for the extraction of graph input features such as the state of our settings. GCN has been utilized to extract features of graph-shaped input in various machine learning studies [22], [23]. Based on GCN, a dynamic action recognition method of human body skeletons was proposed in [22]. By considering the skeleton as a graph wherein the nodes were the joints of the human body, this method adapted

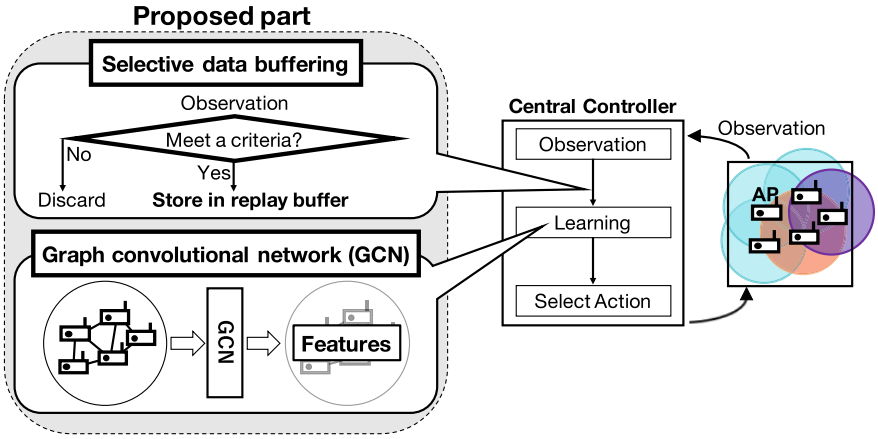



**FIGURE 1.** System model. The central controller observes the environment and selects an action. Each observation is selectively stored in the replay buffer. GCN extracts the features of the information provided by the environment.

GCN to the graph to extract the features of the spatial and temporal actions of the nodes. In [23], a large-scale deep recommendation method was proposed by combining GCN and an efficient random walk approach. The experimental results demonstrated that this method generated higher-quality recommendations than the prior studies in a large-scale graph.

Although a GCN-based channel allocation method can temporarily improve system throughput, the generalization performance decreases over time because of the data duplication in the replay buffer. In WLAN channel allocation problems, once an AP is selected to change its channel, the AP tends to be repeatedly selected to change its channel to the same channel. The imbalanced learning data can advance learning for only the repeatedly observed states and reduce the performance for the other states; this phenomenon is called over-fitting [24]. To prevent the aforementioned degradation in generalization performance, we propose a selective replay buffering that reduces the duplication of the sampling. This idea is based on the undersampling and oversampling approaches for imbalanced learning problems [25].

The contributions of this paper are as follows:

- This paper provides a GCN-based deep reinforcement learning framework that can be applied to problems with an enormous graph-shaped state. Moreover, in the proposed framework, the setting of the optimization objective (i.e., the reward in reinforcement learning) has considerable flexibility as long as it depends on the adjacency relationships of the graph-shaped state.
- This paper proposes a selective replay buffering that is used to avoid the over-fitting caused by the duplication of data for deep learning problems. This approach also functions well for the problem where a certain pair of state and action is repeatedly observed as in WLAN channel allocation problems.
- This paper confirms that the proposed framework successfully increases the cumulative reward. To elaborate, the framework enables a greater reward channel allocation or achieves the optimal channel allocation while reducing the number of changes, compared to the immediate reward maximization method. This is because the immediate reward maximization method does not necessarily achieve the optimal allocation or requires an extended time to achieve the optimal allocation.

The rest of this paper is organized as follows. Section II describes the system model. Section III defines a Markov decision process (MDP) and Section IV introduces the reinforcement learning. Then, Section V introduces the proposed WLAN channel allocation method and Section VI presents an evaluation of the performance of the proposed method. Section VII concludes this study.

## II. SYSTEM MODEL

### A. CHANNEL ALLOCATION PROBLEM IN WIRELESS LANs

In this study we acquire the control algorithm to allocate the optimal channels in the minimum time steps for any initial topology; this is composed of the locations and initial channels of the APs.

Assume that $N$ APs are placed in a square-shaped region and $M$ orthogonal channels with the same bandwidth are available. Let the index set of APs be denoted by $\mathcal{N} = \{1, 2, \ldots, N\}$, and the index set of available channels by $\mathcal{M} = \{1, 2, \ldots, M\}$. In this system model, we do not set specific values for the bandwidths of the channels; rather, we assume that all channels have the same bandwidth without overlapping the frequency bands. The details of the simulation settings are described in Section VI. We regard $c_i \in \mathcal{M}$ as a one-hot vector of $M$ dimensions, (e.g., if AP $i \in \mathcal{N}$ uses Channel $2 \in \mathcal{M}$, then $c_i = [0, 1, 0, \ldots, 0]^{\mathsf{T}}$).

Fig. 1 displays the system model of the deep reinforcement learning-based coordinated WLAN channel allocation. In the proposed system model, a central controller is considered and is responsible for information gathering and channel allocation from/to each AP, as in [1]. More specifically, the central controller observes the communication quality (e.g., the throughput), carrier sensing relationships among the APs, and channels used by the APs at every time step.

Moreover, assume that the central controller can change the channel of an AP at a given time step. The central controller decides what AP changes to what channel based on the deep reinforcement learning from the observation. As indicated in Fig. 1, we propose a new approach to replace the observation buffering part with selective data buffering, and the learning part with the GCN-based approach.

### B. GRAPH STRUCTURE OF STATE

We model the carrier sensing relationships using a contention graph $\mathcal{G} = (\mathcal{N}, \mathcal{E})$. The edges $e_{ij} = \{i, j\} \in \mathcal{E}$ of the graph are connected if and only if the APs $i$ and $j$ are within the carrier sensing range, regardless of their channels. An adjacency matrix is defined as a matrix representation of the graph $\mathcal{G}$, and expressed as an $N \times N$ matrix $\boldsymbol{A} = (A_{ij})$ as follows:

$$A_{ij} = \begin{cases} 1 & \text{if } i \neq j \wedge e_{ij} \in \mathcal{E}, \\ 0 & \text{otherwise.} \end{cases} \tag{1}$$

By focusing on the characteristics of the graph, we analyze the features of the carrier sensing relationships and utilize the relationships to allocate channels based mainly on GCN, which is detailed in Section V-B.

## III. MARKOV DECISION PROCESS

We define an MDP prior to presenting the formulation of the reinforcement learning problem. An MDP is defined as a quadruplet $(\mathcal{S}, \mathcal{A}, \mathbb{P}, \mathcal{R})$: $\mathcal{S}$ is a state space (which denotes a set of states in the environment); $\mathcal{A}$ is an action space (which denotes a set of available actions by the agent); $\mathbb{P}$ is a transition probability to the next state given the current state and action; and $\mathcal{R}$ is a function mapping from a tuple of the current state, action, and following state to a real value (this is called a reward function). The agent is designed to determine the best rule for taking an action for a given state (i.e., policy) using the observed history to date. Then, the agent should transfer to a state that provides a greater reward to itself. Therefore, the state should be adequately associated to the reward, and the method by which the states of an MDP are designed is a critical topic.

In general, the input parameter of the reinforcement learning model is the state. Using the input parameter, the agent selects an action based on the output value (e.g., expected reward) of the learning model and receives a reward with a transition to the next state. This series of events (state input, action selection, state transferring, and reward reception) is performed every time step.

### A. DEFINITION OF MDP FOR WIRELESS LAN CHANNEL ALLOCATION

To design effective states for the WLAN channel allocation problem, the important insight is that the throughput is, in general, significantly influenced by the carrier sensing relationships among the APs. Therefore, we define a state to be a pair of the adjacency matrix and channel vectors for each circumstance, where we define the channel vectors as $M \times N$ matrix $\boldsymbol{C} = [\boldsymbol{c}_1\, \boldsymbol{c}_2\, \ldots\, \boldsymbol{c}_N]$. Note that this state can be considered as a graph signal; thus, we can capture the essential features of the state using GCN. To reduce the number of observable states, we determine an isomorphism between the graphs by comparing their canonical labels, which is detailed in Section III-B.

In a WLAN channel allocation problem, if we define the reward as the total throughput, unfairness could occur, (e.g., the central controller could allocate channels such that certain APs could not transmit. In this study, to improve the overall throughput without such unfairness, we define the reward as the average throughput of the lower 40% APs. Although we define the reward in this manner, this is not an essential constraint. Note that in the proposed system model, because the state is designed based on the carrier sensing relationships among the APs, the reward can be arbitrary as long as it is based on the carrier sensing relationships (e.g., a function of throughputs).

We define the action space $\mathcal{A}$ as the Cartesian product of the indices of the AP and channel $\mathcal{N} \times \mathcal{M}$, where each action $a = (n, m) \in \mathcal{A}$ signifies what AP changes to what channel.

### B. STATE MAPPING METHOD

In WLAN channel allocation problems, if multiple APs are in the same situation, any AP can be selected to change its channel. In such a case, by grouping topologies that are regarded as the same, we can reduce the number of observable states and improve the learning performance.

In this section, we introduce a method to reduce the number of observable states based on canonical labeling [26], [27]. The canonical labels are identical if the graphs exhibit an identical topological structure and identical labeling of the nodes and edges. Thus, by comparing the canonical labels, we sort the graphs in a unique and deterministic manner and consider two graphs as isomorphic if their canonical labels are identical. For the computation of automorphism and canonical labeling of the graphs, we use an open source tool, bliss [28], [29]. Specifically, bliss computes the canonical representative map function $\rho$, wherein the following two conditions are applicable:

- the representative of a graph $\rho(\mathcal{G})$ is isomorphic to graph $\mathcal{G}$.
- the representatives of two graphs, $\rho(\mathcal{G}_1)$ and $\rho(\mathcal{G}_2)$, are identical if and only if the graphs, $\mathcal{G}_1$ and $\mathcal{G}_2$, are isomorphic.

In [28], it is demonstrated that bliss performs canonical labeling.

We also determine the channel indices in a unified manner because the indices of the channels do not influence the system throughput. For example, in Fig. 2, the system throughputs of both graphs are the same regardless of the channel indices. We assign the channel indices in the order of the AP indices after the canonical labeling method.

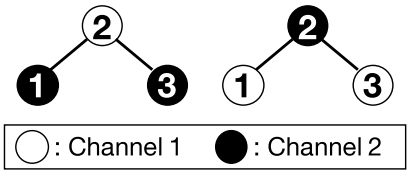



**FIGURE 2.** Two graphs that differ only in indices of channels.

## IV. REINFORCEMENT LEARNING

In this section, we provide an outline of reinforcement learning [30]. Reinforcement learning is a learning problem to acquire the optimal policy. It determines the action for a given state that provides the greatest cumulative reward.

In a reinforcement learning problem, a state value function $V^{\pi}(s)$ of a policy $\pi$ is defined as an expectation of cumulative reward as follows:

$$V^{\pi}(s) = \mathbb{E}_{\pi,\mathbb{P}}\left[\sum_{t=0}^{\infty} \gamma^t r_{t+1} \,\middle|\, s_0 = s\right], \tag{2}$$

where $\gamma \in [0, 1]$ denotes a discount rate, which is the parameter that denotes the value of the future rewards at the current state. Note that the order $\succeq$ between two policies $\pi_1$ and $\pi_2$ is defined as follows:

$$\pi_1 \succeq \pi_2 \overset{\text{def}}{\iff} \forall s \in \mathcal{S},\ V^{\pi_1}(s) \geq V^{\pi_2}(s). \tag{3}$$

Although the order $\succeq$ is not a total order on policy space $\Pi$, it is known that there is at least one optimal (deterministic) policy $\pi^*$, which satisfies $\forall \pi \in \Pi, \pi^* \succeq \pi$, if the reinforcement learning problem is based on an MDP. An optimal policy is commonly learned through the estimation of optimal action-value function, which is written as follows:

$$Q^*(s, a) = \mathbb{E}\left[\mathcal{R}(s, a, s') + \gamma V^{\pi^*}(s')\right]. \tag{4}$$

The goal of reinforcement learning is to obtain an optimal policy $\pi^*$ that maximizes $Q^{\pi}(s, a)$ as follows:

$$\pi^*(a \mid s) = \arg\max_{a \in \mathcal{A}} Q^*(s, a). \tag{5}$$

There are some approaches to obtain the optimal policy $\pi^*(a \mid s)$. Specifically, Q-learning is a method to obtain an optimal policy through the estimation of the optimal action-value function.

## V. PROPOSED SCHEME

In this section, we present the details of the proposed deep reinforcement learning-based method, especially the key ideas, GCN [18]–[21] and selective replay buffering. Because the number of observable AP topologies is extremely high in densely deployed WLANs, we adopt the function approximation of $Q^*(s, a)$. In particular, we use GCN to capture the essential features of the graph signals, which corresponds to the channel information with topologies $(\boldsymbol{A}, \boldsymbol{C})$ in our problem. Furthermore, when an AP selects a channel according to a utility function in common WLANs, a fixed action tends to be selected in certain states, and the duplication of data can cause over-fitting [24]. To prevent the duplication and over-fitting, we introduce the selective replay buffering.

### A. ALGORITHM

In this section, we solve the problem defined in Section II based on deep reinforcement learning. The baseline method is deep Q-network (DQN) [31].

The main features of DQN are experience replay and fixed target Q-network. In general, Q-learning with function approximation could possibly not converge [32]. Fixed target Q-network is a method that promotes convergence by fixing the parameters of the Q-function for a certain period to avoid fluctuations in the target value, which depends on the Q-function itself, when learning the parameters. DQN uses two networks, namely a main network $Q_{\boldsymbol{\theta}}$ (which is the target of the optimization with a weight parameter $\boldsymbol{\theta}$) and a target network $Q_{\boldsymbol{\theta}^-}$ (which is used to calculate the temporal difference errors (TD errors) with a weight parameter $\boldsymbol{\theta}^-$). The parameter of the target network $\boldsymbol{\theta}^-$ is updated to $\boldsymbol{\theta}$ every $I$ time steps and then maintained as fixed between updates. As $I$ increases, the learning becomes more stable while the parameter update frequency decreases. Experience replay is a technique that breaks temporal correlation in the training data. The training data $(s, a, r, s')$ is first stored in a buffer called replay buffer $\mathcal{D}$. Then, DQN updates the parameters using a mini-batch that is constructed using randomly sampled data from the replay buffer. Consequently, there is virtually no time dependence among the data in the mini batch.

In DQN, the parameter $\boldsymbol{\theta}$ is updated in each time step $t$ as follows:

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha\left(Y_t^{\mathrm{Q}} - Q_{\boldsymbol{\theta}}(s_t, a_t)\right)\nabla_{\boldsymbol{\theta}} Q_{\boldsymbol{\theta}}(s_t, a_t), \tag{6}$$

$$Y_t^{\mathrm{Q}} := r_{t+1} + \gamma \max_a Q_{\boldsymbol{\theta}}(s_{t+1}, a). \tag{7}$$

In addition to the original DQN, we employ the following well-known techniques: double DQN (DDQN) [33], dueling network [34], and prioritized experience replay [35], which are known to contribute to the general performance improvement of DQN. DDQN [33] is a DQN-based method to avoid overestimations by employing two different networks. Dueling network [34] is a method that can learn the values of the states without the effect of actions. Prioritized experience replay is an effective data sampling method from the replay buffer $\mathcal{D}$. Details on these methods are presented in the Appendix.

### B. GRAPH CONVOLUTIONAL NETWORKS

In this section, we describe a function approximation method that is suitable for a state designed based on an adjacency matrix. Because the number of observable states is extremely high in the proposed system model, we adopt function approximation to address this large-scale problem.

Feature extraction layers such as a convolution layer [36] play a crucial role in boosting the performance of reinforcement learning, (e.g., AlphaGo [17]). CNN extracts the features of the signals on an input images; however, the input of the proposed system model is not an image; rather, it is a graph. The designed state of the proposed system can be considered as a graph signal; thus, GCN [18]–[21] is

a suitable algorithm to capture the essential features of the state. By applying the GCN layer in the neural network model, we can analyze the graph structure of the APs as a CNN [36] for an input image.

In general, the convolution calculation in the time domain is expressed as the Hadamard product in the frequency domain. Therefore, GCN is expressed by applying an inverse Fourier transformation to the result that corresponds to the Hadamard product after the Fourier transformation. If the input dimension corresponds to $d \in \mathbb{R}$, the following process is adapted to each dimension.

An input vector $\boldsymbol{x} \in \mathbb{R}^N$ is a signal on a graph $\mathcal{G}$ with $N$ nodes. Let $\boldsymbol{D}$ be a degree matrix of the graph, and let $\boldsymbol{L} = \boldsymbol{D} - \boldsymbol{A}$ be its graph Laplacian with the adjacency matrix $\boldsymbol{A}$ of the graph $\mathcal{G}$. Let the graph Laplacian $\boldsymbol{L}$ be orthogonally transformed as $\boldsymbol{L} = U^\top \boldsymbol{x} U$ with eigenvectors $U = (u_1, u_2, \ldots, u_N)$. Subsequently, a graph convolution of input signal $\boldsymbol{x}$ is defined as $\boldsymbol{x} \mapsto U(\theta \odot (U^\top \boldsymbol{x}))$, where $\theta = (\theta_1, \ldots, \theta_N)$ are the parameters to be learnt, and $\odot$ represents the Hadamard product.

### C. SELECTIVE REPLAY BUFFERING

This section describes the proposed selective data buffering applied to the replay buffer $\mathcal{D}$. When the agent selects actions based on a policy that typically has the optimal response, (e.g., $\epsilon$-greedy with tiny $\epsilon$), a fixed action tends to be selected in certain states. As mentioned previously, the imbalanced data can advance learning for only the experienced states and reduce the performance for the inexperienced states, which is called over-fitting [24]. To prevent over-fitting, we propose selective replay buffering, which aims to reduce buffering the same data in the replay buffer. This idea is based on the undersampling and oversampling approaches for imbalanced learning problems [25].

**Algorithm 1** Selective Replay Buffering

**Initialize** $X(s, a) \leftarrow 0, \ \forall s, a$
**for** $t \leftarrow 1$ **to** $W$ **do**
  Observe transition $(s_t, a_t, r_{t+1}, s_{t+1})$
  **if** $X(s_t, a_t) \equiv 0 \pmod{\alpha}$ **then**
    **for** $j \leftarrow 1$ **to** $\beta$ **do**
      **if** replay buffer $\mathcal{D}$ is not full **then**
        Store $(s_t, a_t, r_{t+1}, s_{t+1})$ in $\mathcal{D}$
      **else**
        Replace the oldest data in $\mathcal{D}$ by $(s_t, a_t, r_{t+1}, s_{t+1})$
      **end if**
    **end for**
  **end if**
  $X(s_t, a_t) \leftarrow X(s_t, a_t) + 1$
**end for**

Algorithm 1 displays the flow of the buffering to the replay buffer for each episode. The main part of this algorithm is that the observed transition $(s_t, a_t, r_{t+1}, s_{t+1})$ is stored in replay buffer $\mathcal{D}$ if the transition has never been observed or every $\alpha$ times that the same state transition is observed. Furthermore, to prevent observations remaining in the replay buffer $\mathcal{D}$ for an extended time, we store an observation $\beta$ times repeatedly. Note that $X(s, a)$ is the number of experiences performing an action $a$ from a state $s$, which is initialized at the beginning of each episode. This method reduces the duplication of data stored in replay buffer.

## VI. SIMULATION EVALUATION

In this section, we validate the efficiency of the proposed scheme using proof-of-concept simulations. Assume that the step number of one episode is fixed in these simulations.

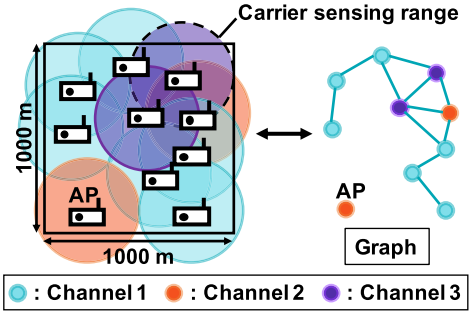



**FIGURE 3.** Simulation environment. Position and channel of each AP is randomly initialized at the beginning of each episode. The environment can be expressed by the graph signal.

Fig. 3 displays the simulation environment. APs are placed on a 1000 m × 1000 m square region. The positions and channels of the APs are randomly initialized at the beginning of each episode. We set the carrier sensing range as 550 m as in [37], [38]; the APs within the carrier sensing range have a carrier sensing relationship. The relationship can be expressed through a graph as in Fig. 3, where the APs and carrier sensing relationships are denoted by nodes and edges, respectively.

Fig. 4 indicates the overall architectures used in the simulations where Figs. 4(a) and 4(b) represent the GCN-based and simple neural network models, which comprise only fully connected layers, respectively. In detail, the "Dense" layer represents the fully connected dense layer; the "Batch Normalization" layer is the function layer which increases the learning speed and restrains the over-fitting [39]; the "ReLU" layer is a well-known activation function [40]; and the "Graph Convolution" layer represents the graph convolutional layer detailed in Section V-B. Note that each graph convolution layer requires that the adjacency matrix consider an input signal as a graph signal. The outputs are the estimated action values $Q(s, a) \ \forall a \in \mathcal{A}$.

The simulation parameters are summarized in Table 2. Assume that the step number of one episode is 500 and the episode number is 10000. The central controller can change the channel of an AP at a given time step. In these simulations, we define the reward, the objective of the optimization, as the average throughput of the lower 40% APs. Note that the setting of the reward has considerable flexibility as long as it

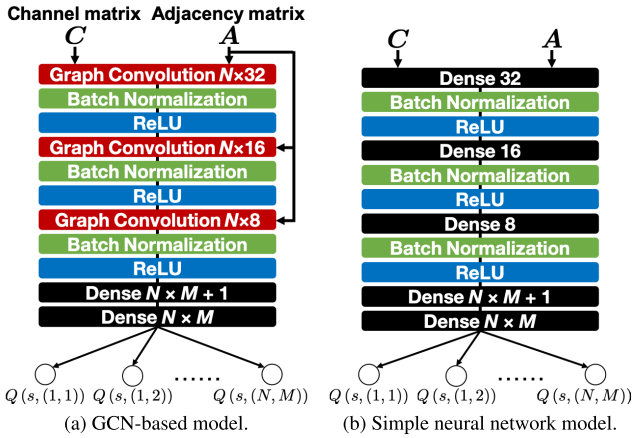



**FIGURE 4.** Structures of GCN-based and simple neural network models. The input is the pair of adjacency matrix $A$ and channel vectors $C$, and the outputs are the estimated action values $Q(s, a)\ \forall a \in \mathcal{A}$.

**TABLE 2.** Simulation parameters.

| | Quantity | Value |
|---|---|---|
| $N$ | Number of APs | 10 |
| $M$ | Number of available channels | 3 |
| | Carrier sensing range | 550 m |
| | Scattering region of APs | 1000 m × 1000 m |
| | Throughput | BoE throughput [37] |
| $r$ | Reward | Average throughput of lower 40% APs |
| $W$ | Step number of each episode | 500 time steps |
| | Episode number of learning | 10000 episodes |
| $I$ | Update period of $Q_{\theta-}$ | 200 episodes |
| $\gamma$ | Discount rate | 0.9 |
| | Batch size | 32 |
| | Optimizer | Adam [41] (learning rate = 0.001) |
| | Loss function | Huber loss |
| $\epsilon$ | Parameter of $\epsilon$-greedy | 0.1 |
| | Replay buffer size | 10000 |
| $\alpha$ | Thinning-rate of observed data | 2 |
| $\beta$ | Repeat number of buffering data | 2 |

depends on the adjacency relationships of the graph-shaped state as already discussed in Section I. Let the $n$th lowest throughput ($n \in \mathcal{N}$) among 10 APs in the $k$th topology ($k \in \{1, 2, \ldots, 10000\}$) be denoted by $\xi_k^{(n)}$. The reward, the average throughput of the lower 40% APs, i.e. 4 APs, is expressed by $\frac{1}{4}\sum_{n=1}^{4}\xi_k^{(n)}$ in the $k$th topology.

To evaluate the generalization performance of the Q-function during learning, we prepared 100 test topologies, where the APs were randomly located and the channel of all the APs were set to Channel 1. When we evaluated the generalization performance, we used the snapshot of the Q-function of that time to select the AP channel to be changed in each time step. For each test topology, the central controller repeated the changing of the channel of an AP according to the output of the Q-function 20 times. We used the reward corresponding to the state after 20 time steps from the initial state as the final reward of the test topology.

For reproducibility of results, we used the back-of-the-envelope (BoE) throughput evaluation technique [37] to model the throughput of the APs according to the carrier sensing relationships among the APs in each channel configuration. The BoE technique allows the adoption of shortcuts in performance evaluation and bypasses complicated stochastic analysis. The BoE throughput was derived under the assumption that each AP had a link with an STA at a given time, and all the links were saturated, i.e., all links always had frames to send. All simulations in this paper followed this assumption. Moreover, the simulations used a normalized throughput of the bandwidth as an observed throughput according to the BoE technique, i.e., the observed throughput had a value between 0 and 1. As a data collecting policy, we used $\epsilon$-greedy [30], which randomly selects an action with probability $\epsilon$ and selects a greedy action with probability $1-\epsilon$. If $\epsilon$ was small, the agent tended to repeatedly select a fixed action in certain states, and the imbalanced stored data caused over-fitting.

### A. EVALUATION OF GENERALIZATION PERFORMANCE FOR RANDOM TOPOLOGY

We compared the following five methods:

- a deep reinforcement learning-based method with the simple neural network model in Fig. 4(b), referred to as "DRL without GCN".
- a deep reinforcement learning-based method with the GCN model in Fig. 4(a), termed as "DRL with GCN".
- a deep reinforcement learning-based method with the GCN model in Fig. 4(a) and selective data buffering explained in Section V-C, denoted as "DRL with GCN and buffer method".
- a random action selection method, referred to as "Random".
- a distributed method based on potential game [10], referred to as "Distributed method".

As mentioned in Section I, as far as we know, none of prior studies addressed to exactly the same problem as this paper. Therefore, as a comparison method, we employed a potential game-based method [10], which allocates channels based on the adjacency relationships of APs to improve system throughput indirectly. In the potential game-based method, the action with the greater payoff function was stochastically selected with higher probability among other choices. This method is guaranteed to achieve the Nash equilibrium [8]. In this paper, we defined the payoff function such that the number of carrier sensing relationships (i.e., network collisions) was minimized according to [10]. In [10], it was proven that minimizing the network collisions provides a near-optimal throughput. Let the channel used by AP $i \in \mathcal{N}$ at time step $t$ be denoted by $c_i[t] \in \mathcal{M}$. At each time step $t$, the probability that AP $i$ selects the next channel $c_i[t+1]$ is expressed as follows:

$$P(c_i[t], c_{-i}[t]) = \frac{\exp[\zeta u_i(c_i[t], \boldsymbol{c}_{-i}[t])]}{\sum_{c_i' \in \mathcal{M}} \exp[\zeta u_i(c_i'[t], \boldsymbol{c}_{-i}[t])]}, \tag{8}$$

$$u_i(\boldsymbol{c}) := -\sum_{j \neq i} \mathbb{1}(c_j = c_i)\, \mathbb{1}(e_{ij} \in \mathcal{E}), \tag{9}$$

$$\boldsymbol{c}_{-i}[t] := (c_1[t], \ldots, c_{i-1}[t], c_{i+1}[t], \ldots, c_N[t]), \tag{10}$$

where $u_i(\boldsymbol{c})$ denotes a payoff function; $\mathbb{1}(x)$ denotes an indicator function that is one if event $x$ is true and is zero otherwise; and $\zeta \geq 0$ denotes the parameter that determines the degree of selecting the state with a high payoff function. In this simulation, the parameter $\zeta$ was set to 0.1.

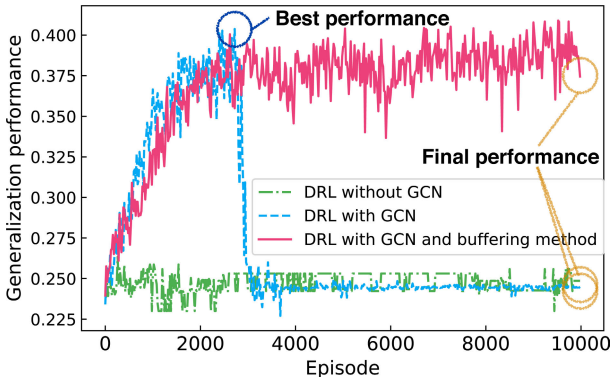



**FIGURE 5.** Transitions of mean reward of 100 test topologies after performing 20-step greedy actions.

Fig. 5 displays the learning curves representing the transitions of the generalization performance evaluated every 20 episodes. We evaluated the generalization performance by allocating channels for 100 inexperienced test topologies according to the learning models and observed the rewards after 20-step greedy actions. Each value in Fig. 5 indicates the mean reward of 100 test topologies after performing 20-step greedy actions. The generalization performances of the methods using GCN-based model increased when the learning progressed, whereas that of the simple neural network models exhibited virtually no increase. However, the generalization performance of the method without selective data buffering using the GCN-based model decreased after a certain amount of time. This is because the model learned for experienced states and reduced the performance for inexperienced states, which is called over-fitting. By employing selective data buffering, we could maintain the generalization performance at a high level.

Fig. 6 displays the cumulative distribution functions (CDFs) of the rewards of 100 test topologies at the best and final performance points in Fig. 5. As indicated in the figure, the proportions of the high reward state of the results of the deep reinforcement learning-based methods with GCN-based model exceeded those of the other methods. Therefore, using the GCN-based model, the learning performance exceeded that of the simple neural network model. Moreover, the effect of the over-fitting can be seen in this figure by comparing the "DRL with GCN (best)" and "DRL with GCN (final)" lines. We can observe that the over-fitting was avoided by employing the proposed selective data buffering.

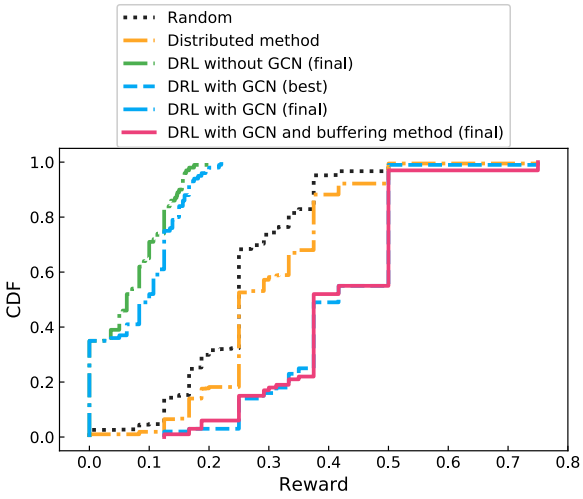



**FIGURE 6.** CDFs of rewards of 100 test topologies.

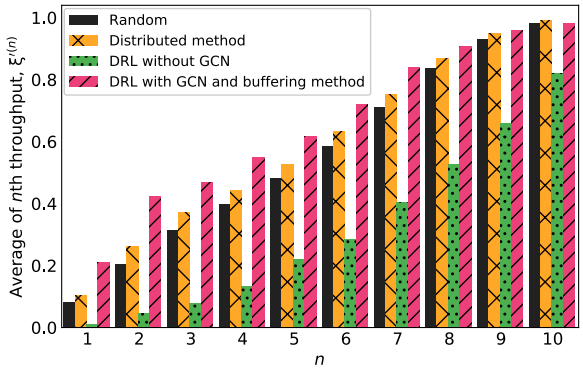



**FIGURE 7.** Averages of the $n$th lowest throughput.

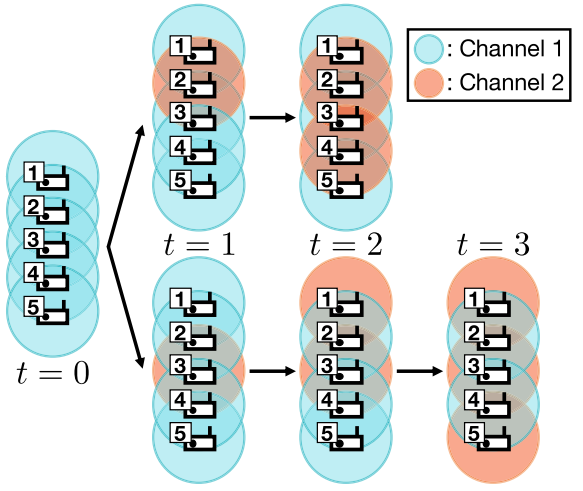
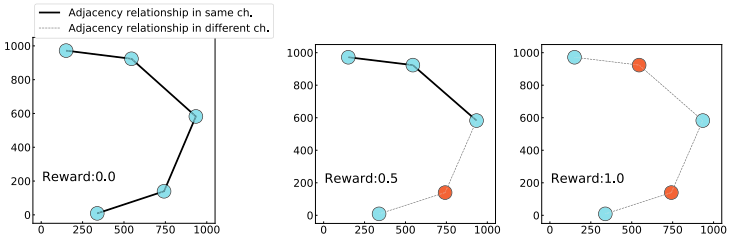



**FIGURE 8.** Channel allocation sequence towards equivalent allocation. The upper sequence is more desirable because the time steps required for achieving the optimal channel allocation is shorter, and thus the cumulative reward is greater.

Let the $n$th lowest throughput ($n \in \mathcal{N}$) among 10 APs in the $l$th test topology ($l \in \{1, 2, \ldots, 100\}$) be denoted by $\xi_l^{\prime(n)}$. The averages of the $n$th lowest throughputs in 100 test topologies $\xi^{\prime(n)} := \frac{1}{100}\sum_{l=1}^{100} \xi_l^{\prime(n)}$ are displayed in Fig. 7. This figure indicates that the averages of the $n$th lowest




(a) $t = 0$. (b) $t = 1$. (c) $t = 2$.

**FIGURE 9. Channel allocation sequence based on proposed method. The reward is the average throughput of the lower two APs of the five APs. The cumulative reward is maximized in the least number of time steps.**

throughput $\xi'^{(n)}$ of the deep reinforcement learning-based method with selective data buffering using the GCN-based model was greater than those of the other methods. In particular, the first to fourth lowest throughputs $\xi'^{(n)}(n = \{1, 2, 3, 4\})$ increased by using the proposed method. This is because we defined the reward of the learning as the average of the lowest four throughputs $\frac{1}{4}\sum_{n=1}^{4}\xi'^{(n)}$. Moreover, by comparing the deep reinforcement learning-based methods with and without GCN, we confirmed that GCN makes it possible to increase the performance through training.

The performances of the potential game-based method displayed in Figs. 6 and 7 are inferior to those of the proposed method. This can be attributed, possibly, to the fact that the main target of the potential game-based method was not to improve the system throughput directly; rather it was to reduce the number of carrier sensing relationships, which could influence the system throughput.

### *B. EVALUATION OF CHANNEL ALLOCATION SEQUENCE*

In this section, we evaluated the efficiency of the proposed method to maximize the cumulative reward. Specifically, we evaluated the channel allocation sequence from two perspectives: 1) how fast the proposed method achieved a destination (convergence speed perspective), where a method converging faster is superior, and 2) how proactively the proposed method selected channels (delayed reward perspective), where a method converging to a channel configuration with a greater reward is superior. The convergence speed perspective is evaluated in Section VI-B1; the delayed reward perspective is evaluated in Section VI-B2.

#### 1) CONVERGENCE SPEED PERSPECTIVE

The channel allocation sequence influences the system throughput during the control, even if the destinations are the same. Fig. 8 is a hypothetical example that can be used for explanation. We consider a case where the numbers of APs and channels are five and two, respectively. The positions of the APs are indicated in Fig. 8, where each AP has relationships only with its adjacent APs. The optimal channel allocation is to allocate Channel 1 (or Channel 2) to APs 1, 3, and 5, and Channel 2 (or Channel 1) to APs 2 and 4, respectively, i.e., we should allocate channels so as to not allocate the same channel to adjacent APs. In this case, we can use two channel allocation sequences for the optimal allocation as indicated in Fig. 8. One sequence is to change the channels of APs 2 and 4, and another is to change the channels of APs 1, 3, and 5. The former sequence requires two time steps, whereas the latter requires three time steps. For improving the system throughput even during the channel allocation process, the former sequence is more desirable. Therefore, we aim to achieve the optimal channel allocation in fewer time steps.

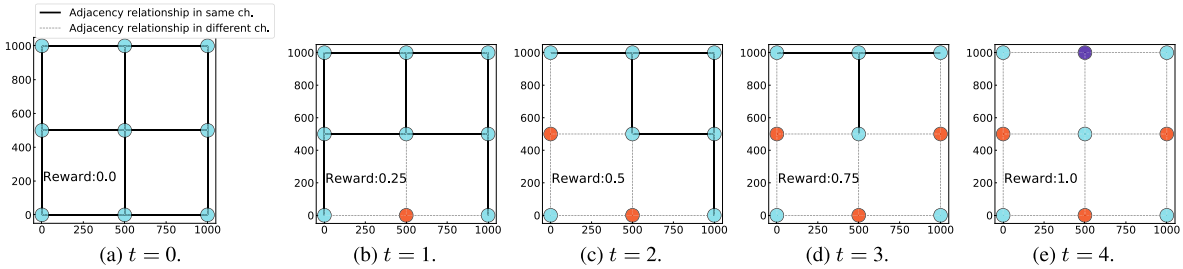
Fig. 9 displays the channel allocation sequence of a test topology where the numbers of APs and channels are five and two, respectively. In these figures, the nodes represent the APs and the edges represent the adjacency relationships. The solid line indicates the contention among the APs using the same channel within the carrier sensing range; the dashed line connects the APs using different channels within the carrier sensing range. The colors of the nodes indicate the channels used by the APs, and blue and orange denote Channels 1 and 2, respectively. The topology of this figure is as that of Fig. 8. From this figure, we can observe that the proposed method can allocate channels in the desirable sequence with the maximum cumulative reward. Similarly, Fig. 10 displayed the channel allocation sequence of a test topology where the numbers of APs and channels are nine and three, respectively. In addition to Channel 1 and 2, the purple nodes denote APs using Channel 3. In this topology, in the optimal channel allocation, orthogonal channels are allocated to adjacent APs. The proposed method can allocate channels in the desirable sequence with the maximum cumulative reward.

It is remarkable that in each figure, the channel of the AP centered in the topology is not changed at the first step $t = 1$. Note that this change does occur in the case of the immediate reward maximization. The immediate reward maximization method is a method that maximizes the immediate reward received at each time step, whereas the proposed method maximizes the cumulative reward in the channel allocation sequence.

#### 2) DELAYED REWARD PERSPECTIVE

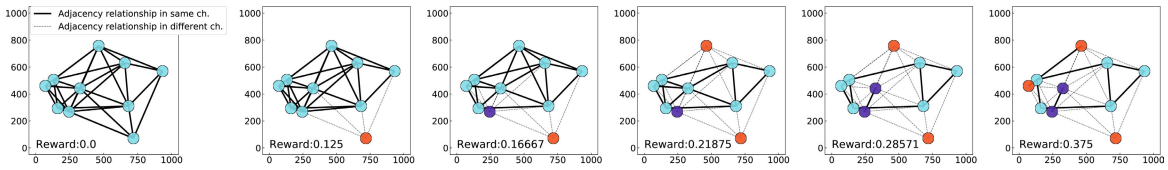
Moreover, we can demonstrate that the proposed method is superior to the immediate reward maximization in performance at the end of the sequence.

Figs. 11 and 12 display the channel allocation sequences of one test topology based on the immediate reward maximization and the proposed methods, respectively.



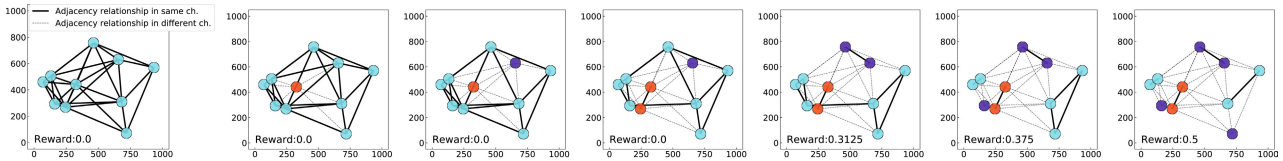

(a) $t = 0$. (b) $t = 1$. (c) $t = 2$. (d) $t = 3$. (e) $t = 4$.

**FIGURE 10.** Channel allocation sequence based on proposed method. The reward is the average throughput of the lower four APs of the nine APs. The cumulative reward is maximized in the least number of time steps.




(a) $t = 0$. (b) $t = 1$. (c) $t = 2$. (d) $t = 3$. (e) $t = 4$. (f) $t = 5$.

**FIGURE 11.** Channel allocation sequence based on immediate reward maximization. The reward is the average throughput of the lower four APs of 10 APs.




(a) $t = 0$. (b) $t = 1$. (c) $t = 2$. (d) $t = 3$. (e) $t = 4$. (f) $t = 5$. (g) $t = 6$.

**FIGURE 12.** Channel allocation sequence based on proposed method. The proposed method can achieve superior channel allocation compared to immediate reward maximization. The reward is the average throughput of the lower four APs of 10 APs.

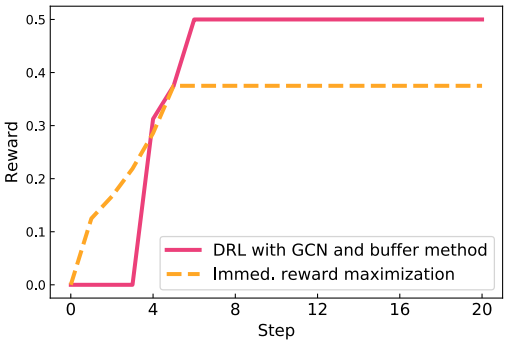



**FIGURE 13.** Reward transitions based on proposed method and immediate reward maximization method. The final reward of the proposed method is greater than that of the immediate reward maximization method.

The numbers of APs and channels are 10 and three, respectively. Furthermore, Fig. 13 displays the time series of the reward transitions of each method in this example. The final reward of the proposed method is 0.5, whereas that of the immediate reward maximization method is 0.375. In the immediate reward maximization method, the agent takes an action that maximizes the immediate reward at each time step; thus, this method can fall into the local optimal allocation. To achieve the global optimal allocation using this method, plural APs must change their channels simultaneously. Because we can select only one AP to change at each time step in this simulation, this method cannot achieve the global optimal allocation.

Conversely, the proposed method achieves the global optimal allocation with maximum reward. The rewards are zero until $t = 3$ for the proposed method. This result indicates that the proposed method selects the action that maximizes the cumulative reward, which includes the delayed rewards received after channel allocation at each time step, even if the immediate reward is small. This can be attributed to the fact that the reinforcement learning updates the learning parameters for maximizing the cumulative reward.

## VII. CONCLUSION

A deep reinforcement learning-based channel allocation scheme was proposed for densely deployed WLANs. First, to capture the essential features of the carrier sensing relationships among the APs, we applied GCN to a graph where the APs were connected within their carrier sensing ranges. Then, we proposed a selective data buffering to prevent over-fitting by reducing the duplication of the sampling data specific to WLAN channel allocation problems. The main learning algorithm of this method was DDQN employing dueling network and prioritized experience replay. Furthermore, because the number of observable states was extremely high in this problem, we used canonical labeling to reduce the number and improve the learning performance. The simulation results indicated that the proposed scheme achieved greater rewards after 20-step greedy actions from a given initial state than the compared methods. Using GCN, we improved the learning performance compared with that of the simple neural network

model, which comprised only fully connected layers. Finally, we demonstrated that the proposed method could allocate channels in a short number of time steps and with a large cumulative reward.

## APPENDIX

### A. DDQN

DDQN [33] is a DQN-based method to avoid over-fitting. DDQN uses two networks in the same manner as DQN as discussed in Section IV, i.e., $Q_{\boldsymbol{\theta}}$ and $Q_{\boldsymbol{\theta}^-}$. The error value of DDQN is expressed as follows:

$$Y_t^{\mathrm{DDQN}} := r_{t+1} + \gamma Q_{\boldsymbol{\theta}^-}(s_{t+1}, \arg\max_a Q_{\boldsymbol{\theta}}(s_{t+1}, a)).$$

The parameters are updated to minimize this error value.

### B. DUELING NETWORK

Dueling network [34] is a method that can learn what states are (or are not) valuable without having to learn the effect of each action for each state. Dueling network includes two streams to separately estimate the state-value and advantages for each action in a neural network architecture. The output value corresponds to the total value of the two streams.

### C. PRIORITIZED EXPERIENCE REPLAY

In this paper, we sample the training data from the replay buffer $\mathcal{D}$ according to the prioritized experience replay [35], which allocates priority to all samples based on the TD errors. The TD error $\delta_t$ in DDQN is expressed as follows:

$$\delta_t = r_{t+1} + \gamma Q_{\boldsymbol{\theta}^-}(s_{t+1}, \arg\max_a Q_{\boldsymbol{\theta}}(s_{t+1}, a)) - Q_{\boldsymbol{\theta}^-}(s_t, a_t).$$

The probability of selecting sample $i$ is expressed as follows:

$$P(i) = \frac{p_i^\lambda}{\sum_k p_k^\lambda}, \tag{11}$$

$$p_i = |\delta_i| + \mu_0, \tag{12}$$

where $\mu_0$ is a small positive number that prevents the sampling probabilities from being zero when the TD error is zero. $\lambda$ is a parameter that determines the degree of prioritizing for sampling (in particular, when $\lambda = 0$, the sampling is uniformly randomly implemented). If the absolute value of the TD error $|\delta_t|$ is relatively large in the replay buffer, the probability of selecting the corresponding sample increases.

## ACKNOWLEDGMENT

This article was presented in part at the 2019 IEEE Vehicular Technology Conference-Fall.

## REFERENCES

[1] S. Chieochan, E. Hossain, and J. Diamond, "Channel assignment schemes for infrastructure-based 802.11 WLANs: A survey," *IEEE Commun. Surveys Tutr.*, vol. 12, no. 1, pp. 124–136, 2010.

[2] *Status of Project IEEE P802.11be*. Accessed: Nov. 20, 2019. [Online]. Available: http://www. ieee802.org/11/Reports/tgbe_update.htm

[3] J. Riihijarvi, M. Petrova, P. Mahonen, and J. D. A. Barbosa, "Performance evaluation of automatic channel assignment mechanism for IEEE 802.11 based on graph colouring," in *Proc. IEEE PIMRC*, Helsinki, Finland, Sep. 2006, pp. 1–5.

[4] J. Chen, G. De Veciana, and T. Rappaport, "Site-specific knowledge and interference measurement for improving frequency allocations in wireless networks," *IEEE Trans. Veh. Technol.*, vol. 58, no. 5, pp. 2366–2377, 1st Quart., 2009.

[5] M. Haidar, R. Ghimire, H. Al-Rizzo, R. Akl, and Y. Chan, "Channel assignment in an IEEE 802.11 WLAN based on signal-to-interference ratio," in *Proc. Can. Conf. Electr. Comput. Eng.*, May 2008.

[6] C. Kai, Y. Liang, T. Huang, and X. Chen, "To bond or not to bond: An optimal channel allocation algorithm for flexible dynamic channel bonding in WLANs," in *Proc. IEEE 86th Veh. Technol. Conf. (VTC-Fall)*, Sep. 2017, pp. 1–6.

[7] A. Raschella, M. Mackay, F. Bouhafs, and B. I. Teigen, "Evaluation of channel assignment algorithms in a dense real world WLAN," in *Proc. 4th Int. Conf. Comput., Commun. Secur. (ICCCS)*, Oct. 2019, pp. 1–5.

[8] D. Monderer and L. S. Shapley, "Potential games," *Games Econ. Behav.*, vol. 14, no. 1, pp. 124–143, May 1996.

[9] K. Yamamoto, "A comprehensive survey of potential game approaches to wireless networks," *IEICE Trans. Commun.*, vol. E98.B, no. 9, pp. 1804–1823, 2015.

[10] Y. Xu, J. Wang, Q. Wu, A. Anpalagan, and Y.-D. Yao, "Opportunistic spectrum access in cognitive radio networks: Global optimization using local interaction games," *IEEE J. Sel. Topics Signal Process.*, vol. 6, no. 2, pp. 180–194, Apr. 2012.

[11] S. I. Suliman, G. Kendall, and I. Musirin, "Artificial immune algorithm in solving the channel assignment task," in *Proc. IEEE Int. Conf. Control Syst., Comput. Eng. (ICCSCE)*, Nov. 2014, pp. 153–158.

[12] B. S. Ghahfarokhi, "Distributed QoE-aware channel assignment algorithms for IEEE 802.11 WLANs," *Wireless Netw.*, vol. 21, no. 1, pp. 21–34, Jan. 2015.

[13] K. S. Narendra and M. A. Thathachar, *Learning Automata: An Introduction*. Upper Saddle River, NJ, USA: Prentice-Hall, May 1989.

[14] O. Jeunen, P. Bosch, M. Van Herwegen, K. Van Doorselaer, N. Godman, and S. Latré, "A machine learning approach for IEEE 802.11 channel allocation," in *Proc. IEEE CNSM* , Roma, Italy, Nov. 2018, pp. 28–36.

[15] N. C. Luong, D. T. Hoang, S. Gong, D. Niyato, P. Wang, Y.-C. Liang, and D. I. Kim, "Applications of deep reinforcement learning in communications and networking: A survey," *IEEE Commun. Surveys Tutr.*, vol. 21, no. 4, pp. 3133–3174, 4th Quart., 2019.

[16] 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.

[17] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, and D. Hassabis, "Mastering the game of Go with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, Jan. 2016.

[18] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," 2013, *arXiv:1312.6203*. [Online]. Available: https://arxiv.org/abs/1312.6203

[19] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proc. Adv. Neural Inf. Process. Syst.*, Barcelona, Spain, Dec. 2016, pp. 3844–3852.

[20] M. Henaff, J. Bruna, and Y. LeCun, "Deep convolutional networks on graph-structured data," Jun. 2015, *arXiv:1506.05163*. [Online]. Available: https://arxiv.org/abs/1506.05163

[21] F. Scarselli, M. Gori, A. Chung Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Trans. Neural Netw.*, vol. 20, no. 1, pp. 61–80, Jan. 2009.

[22] S. Yan, Y. Xiong, and D. Lin, "Spatial temporal graph convolutional networks for skeleton-based action recognition," in *Proc. AAAI*, New Orleans, LA, USA, Feb. 2018.

[23] R. Ying, R. He, K. Chen, P. Eksombatchai, W. L. Hamilton, and J. Leskovec, "Graph convolutional neural networks for Web-scale recommender systems," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2018, pp. 974–983.

[24] C. Zhang, O. Vinyals, R. Munos, and S. Bengio, "A study on overfitting in deep reinforcement learning," 2018, *arXiv:1804.06893*. [Online]. Available: https://arxiv.org/abs/1804.06893

[25] H. He and E. A. Garcia, "Learning from imbalanced data," *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 9, pp. 1263–1284, Sep. 2009.

[26] B. D. McKay, "Practical graph isomorphism," Ph.D. dissertation, Dept. Comput. Sci., Vanderbilt Univ., Nashville, TN, USA, 1981, vol. 30.

[27] M. Kuramochi and G. Karypis, "Frequent subgraph discovery," in *Proc. IEEE Int. Conf. Data Mining*, Nov. 2002, pp. 313–320.
[28] T. Junttila and P. Kaski, "Engineering an efficient canonical labeling tool for large and sparse graphs," in *Proc. SIAM ALENEX*, Astor Crowne Plaza, NO, Lousiana, Jan. 2007, pp. 135–149.
[29] J. Tommi and K. Petteri, *Bliss: A Tool for Computing Automorphism Groups and Canonical Labelings of Graphs*. Accessed: Nov. 20, 2019. [Online]. Available: http://www.tcs.hut.fi/Software/bliss/
[30] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*, 2nd ed. Cambridge, MA, USA: MIT Press, Jan. 2018.
[31] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, Feb. 2015.
[32] S. Adam, L. Busoniu, and R. Babuska, "Experience replay for real-time reinforcement learning control," *IEEE Trans. Syst., Man, Cybern. C, Appl. Rev.*, vol. 42, no. 2, pp. 201–212, Mar. 2012.
[33] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in *Proc. AAAI*, vol. 2, Phoenix, Arizona, USA, Feb. 2016, pp. 2094–2100.
[34] Z. Wang, T. Schaul, M. Hessel, H. Van Hasselt, M. Lanctot, and N. De Freitas, "Dueling network architectures for deep reinforcement learning," 2015, *arXiv:1511.06581*. [Online]. Available: https://arxiv.org/abs/1511.06581
[35] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," in *Proc. ICLR*, May 2016.
[36] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. NIPS*, Lake Tahoe, NV, USA, Dec. 2012, pp. 1097–1105.
[37] S. Chang Liew, C. Hong Kai, H. Ching Leung, and P. Wong, "Back-of-the-envelope computation of throughput distributions in CSMA wireless networks," *IEEE Trans. Mobile Comput.*, vol. 9, no. 9, pp. 1319–1331, Sep. 2010.
[38] M. Durvy, O. Dousse, and P. Thiran, "Self-organization properties of CSMA/CA systems and their consequences on fairness," *IEEE Trans. Inf. Theory*, vol. 55, no. 3, pp. 931–943, Mar. 2009.
[39] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015, *arXiv:1502.03167*. [Online]. Available: https://arxiv.org/abs/1502.03167
[40] V. Nair and G. E. Hinton, "Rectified linear units improve restricted Boltzmann machines," in *Proc. ICML*, Haifa, Israel, Jun. 2010, pp. 807–814.
[41] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: https://arxiv.org/abs/1412.6980

**KOTA NAKASHIMA** (Student Member, IEEE) received the B.E. degree in electrical and electronic engineering from Kyoto University, in 2018, where he is currently pursuing the M.I. degree with the Graduate School of Informatics. He received the VTS Japan Young Researcher's Encouragement Award, in 2018.

**SHOTARO KAMIYA** (Student Member, IEEE) received the B.E. degree in electrical and electronic engineering from Kyoto University, in 2015, and the M.E. degree from the Graduate School of Informatics, Kyoto University, in 2017, where he is currently pursuing the Ph.D. degree. He is a member of the IEICE. From 2017 to 2019, he was a Research Fellow (DC1) of the Japan Society for the Promotion of Science (JSPS). He received the TELECOM System Technology Award for Students, the Outstanding Paper Award for Young C&C Researchers, and the IEEE Kansai Section Student Award, in 2017, 2019, and 2019, respectively.

**KAZUKI OHTSU** (Student Member, IEEE) received the B.E. degree in electrical and electronic engineering from Kyoto University, in 2018. He is currently pursuing the M.E. degree with the Graduate School of Informatics, Kyoto University.

**KOJI YAMAMOTO** (Member, IEEE) received the B.E. degree in electrical and electronic engineering and the master's and Ph.D. degrees in informatics from Kyoto University, in 2002, 2004, and 2005, respectively. From 2008 to 2009, he was a Visiting Researcher with WirelessKTH, Royal Institute of Technology (KTH), Sweden. Since 2005, he has been with the Graduate School of Informatics, Kyoto University, where he is currently an Associate Professor. His research interests include radio resource management, game theory, and machine learning. He is a Senior Member of the IEICE and a member of the Operations Research Society of Japan. From 2004 to 2005, he was a Research Fellow of the Japan Society for the Promotion of Science (JSPS). He serves as the Track Co-Chair for APCC 2017, CCNC 2018, APCC 2018, and CCNC 2019, and the Vice Co-Chair for IEEE ComSoc APB CCC. He was a Tutorial Lecturer in ICC 2019. He received the PIMRC 2004 Best Student Paper Award, in 2004, and the Ericsson Young Scientist Award, in 2006. He also received the Young Researcher's Award, the Paper Award, the SUEMATSU-Yasuharu Award from the IEICE of Japan, in 2008, 2011, and 2016, respectively, and the IEEE Kansai Section GOLD Award, in 2012. He serves as an Editor for the IEEE Wireless Communications Letters and the *Journal of Communications and Information Networks*.

**TAKAYUKI NISHIO** (Member, IEEE) received the B.E. degree in electrical and electronic engineering from Kyoto University, Kyoto, Japan, in 2010, and the master's and Ph.D. degrees in communications and computer engineering from the Graduate School of Informatics, Kyoto University, in 2012 and 2013, respectively. Since 2013, he has been an Assistant Professor of communications and computer engineering with the Graduate School of Informatics, Kyoto University. From 2012 to 2013, he was a Research Fellow (DC1) of the Japan Society for the Promotion of Science (JSPS).

**MASAHIRO MORIKURA** (Member, IEEE) received the B.E., M.E., and Ph.D. degrees in electronics engineering from Kyoto University, Kyoto, Japan, in 1979, 1981, and 1991, respectively. In 1981, he joined NTT, where he was engaged in the research and development of TDMA equipment for satellite communications. From 1988 to 1989, he was a Guest Scientist with the Communications Research Centre, Canada. From 1997 to 2002, he was active in the standardization of the IEEE 802.11a-based wireless LAN. He is currently a Professor with the Graduate School of Informatics, Kyoto University. His current research interests include WLANs and M2M wireless systems. He received the Paper Award and the Achievement Award from IEICE, in 2000 and 2006, respectively, the Education, Culture, Sports, Science and Technology Minister Award, in 2007, the Maejima Award, in 2008, and the Medal of Honor with Purple Ribbon from Japan's Cabinet Office, in 2015.