Reducing Transmission Delay in EDCA Using Policy Gradient Reinforcement Learning

Masao Shinzai*, Yusuke Koda*, Koji Yamamoto*†, Takayuki Nishio*, and Masahiro Morikura*
* Graduate School of Informatics, Kyoto University, Yoshida-honmachi, Sakyo-ku, Kyoto 606-8501 Japan
† kyamamot@i.kyoto-u.ac.jp

***Abstract*—Towards ultra-reliable and low-latency communications, this paper proposes a packet mapping algorithm in an enhanced distributed channel access (EDCA) scheme using policy gradient reinforcement learning (RL). The EDCA scheme provides higher priority packets with more transmission opportunities by mapping packets to a predefined access category (AC); thereby, the EDCA scheme supports a higher quality of service in wireless local area networks. In this paper, it is noted that by mapping high priority packets to lower priority ACs, the one-packet delay of a high priority packet can be reduced. In contrast, the mapping algorithm cannot minimize the multiple-packets delay because the mapping algorithm is based on the current status. This is because, from a long-term perspective, mapping high priority packets is required as a countermeasure for collisions, to minimize the multiple-packets delay. As a solution, this paper proposes a new mapping algorithm using RL because RL is suitable for maximizing the reward from a long-term perspective. The key idea is to design the state such that the state involves the number of packets having arrived at each AP in the past, which is an indicator expressing past status. In the designed RL task, the reward, i.e., the multiple-packets delay depends on an overall sequence of states and actions; hence, the recursive value function-based RL algorithms are not compatible. To solve this problem, this paper utilizes policy gradient RL, which learns the packet mapping policy from an overall state-action sequence and a consequent multiple-packets delay. The simulation result reveals that the transmission delay of the proposed mapping algorithm is 13.8% shorter than that of the conventional EDCA mapping algorithm.**

I. Introduction

IEEE 802.11e [1] is specified to support the quality of service (QoS) in wireless local area networks. In IEEE 802.11e, the enhanced distributed channel access (EDCA) scheme is designed to provide prioritized QoS by enhancing the contention-based distributed coordination function. In the EDCA scheme, when packets arrive at an access point (AP), the packets are mapped to the access categories (ACs) based on the priority of the packets. The ACs to which high priority packets are mapped, obtain more transmission opportunities than do the ACs to which low priority packets are mapped.

However, mapping high priority packets to the AC defined in the EDCA scheme is not optimal for minimizing the one-packet delay, which is the interval between when one high priority packet arrives and when the packet transmission is finished. There are situations in which the proportion of high priority packets (in relation to all the packets) is higher than that of the packets with other priorities [2]. In such a situation, by mapping packets based on the EDCA scheme, many high priority packets wait in the queue of the AC to which the high priority packets are mapped. Therefore, by mapping high priority packets to lower priority ACs, the one-packet delay can be reduced compared to the conventional EDCA scheme [3].

However, the mapping algorithms in [3]–[6] cannot minimize the multiple-packets delay. The multiple-packets delay is the interval between when the first high priority packet arrives and when all the packets of all APs finish transmission, in a situation where multiple APs transmit a certain number of high priority packets. APs using the mapping algorithms can obtain more transmission opportunities than APs using the conventional EDCA scheme. However, one AP obtaining many transmission opportunities prevents the other APs from obtaining transmission opportunities due to packet collisions. Therefore, an AP that has obtained more transmission opportunities has to give up transmission opportunities so that other APs can obtain more transmission opportunities. The problem here is that which AP has more transmission opportunities is not judged by the mapping algorithms, which judgment is based on the number of packets waiting in the high priority queue. This is because the number of packets waiting in the high priority queue is an indicator expressing current status.

To solve this problem, we designed a new mapping algorithm to map high priority packets from a long-term perspective. To realize such a mapping algorithm, because the number of high priority packets having arrived in the past is an indicator expressing past status, we designed the new mapping algorithm based on information about all APs, involving the number of high priority packets that arrived previously.

We propose a new mapping algorithm using policy gradient RL to minimize the multiple-packets delay. To take into account the information about all APs, we assume the central controller that maps packets that arrive at each AP to ACs. In such a situation, policy gradient RL is suitable to obtain the optimal mapping algorithm for two reasons. First, because the central controller considers information about other APs, the central controller has to deal with larger state space due to the increase of information about APs. Second, to minimize the multiple-packets delay, the central controller has to map packets from a long-term perspective.

The main contributions of this paper are summarized as follows:

- We propose an RL-based mapping algorithm that mapping high priority packets from a long-term perspective and thereby minimize the multiple-packets delay. The key

idea is to design the state such that the state involves both the current queue status of the ACs in each AP and the number of packets having arrived at each AP. The mapping algorithm in [3] cannot reduce the multiple-packets delay because that mapping algorithm is based on the current status. To solve this problem, we designed a new mapping algorithm based on the number of packets having arrived at an AP in the past, which is an indicator expressing past status. Moreover, we used RL because RL is suitable for maximizing the reward from a long-term perspective.

- We propose to employ policy gradient RL to learn the packet mapping policy highlighting that the designed decision process is a non-Markov decision process (MDP). In detail, the reward i.e., the multiple-packets delay depends not only on a current state-action pair but also upon an overall sequence of states and actions. Thus, recursive value-function-based RL algorithms are not compatible to reduce the multiple-packets delay. Policy gradient RL learns the mapping policy from overall state-action sequences and consequent multiple-packets delay, which is rather compatible with the designed RL task.

The rest of this paper is organized as follows. In Section II, we present a system model. In Section III, we formulate a system model as an RL task. In Section IV, we perform a simulation evaluation of the proposed mapping algorithm. Finally, we present our conclusions in Section V.

II. System Model

Fig. 1 shows a model of the system. In the downlink between AP k and station (STA) k for $k = 1, 2$, the two APs transmit packets using the EDCA scheme. The central controller maps VO packets which arrived at APs to ACs. For simplicity, we consider only two ACs, which are AC voice (AC_VO) and AC video (AC_VI). VO packets are packets mapped to AC_VO by the conventional EDCA scheme, and VO queues are queues where VO packets wait. Similarly, VI packets are packets mapped to AC_VI by the conventional EDCA scheme, and VI queues are queues where VI packets wait.

Fig. 2 shows the mapping control proposed for this system. The average arrival rate of packets is constant and follows a Poisson distribution. VO packets can be mapped to AC_VO and AC_VI. On the other hand, VI packets are always mapped to AC_VI.

The purpose of the work reported in this paper is to provide a mapping algorithm that minimizes the expected value of the transmission delay, which we define as the time between when the first VO packet arrives at one of the APs, to when all APs have finished transmitting N VO packets.

III. Mapping Algorithm by RL

This chapter describes how to minimize the expected value of transmission delay using RL [7]. To formulate the RL task in this system, we define states, actions, and reward.

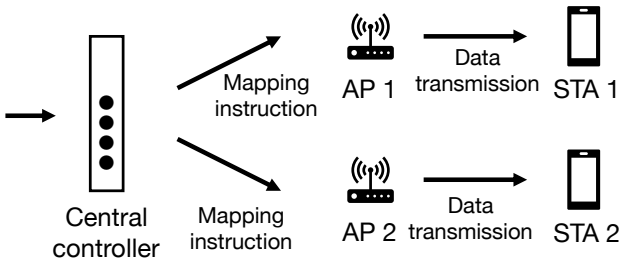

Fig. 1. System model.

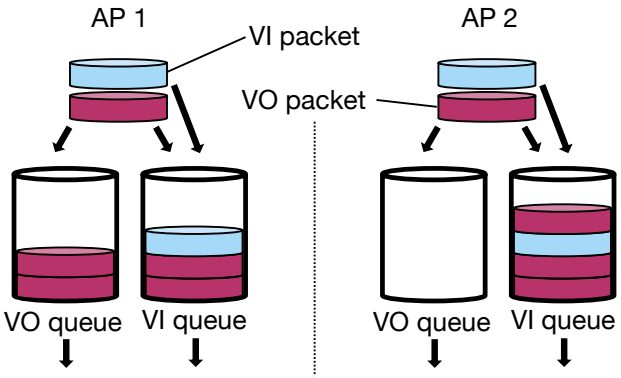

Fig. 2. Mapping control.

We used policy gradient RL to enhance the reward because the RL task is not an MDP. The reward of the RL task depends on an overall sequence of states and actions. In RL methods based on an MDP, the immediate reward is given only by the current state, the current action, and the next state. Recursive value-function-based RL algorithms are not compatible to enhance a reward that depends on an overall sequence of states and actions. However, among RL methods, policy gradient RL can learn the mapping policy from an overall state-action sequence and a consequent multiple-packets delay [8].

We obtained an update equation for a parameter $\boldsymbol{\theta}$ in this system and designed a policy parameterized with $\boldsymbol{\theta}$. In policy gradient RL, the policy is parameterized with a parameter $\boldsymbol{\theta}$. In addition, by updating the parameter $\boldsymbol{\theta}$, the policy is updated to enhance the reward.

A. State, Action, and Reward

We define the state set $\mathcal{S}$ as follows:

$$\begin{aligned} \mathcal{S} &\coloneqq \mathcal{S}_x \times \mathcal{S}_{\mathrm{AP1}} \times \mathcal{S}_{\mathrm{AP2}}, \\ \mathcal{S}_{\mathrm{AP}k} &= \mathcal{S}_A \times \mathcal{S}_{Q_{\mathrm{VO}}} \times \mathcal{S}_{Q_{\mathrm{VI}}} \times \mathcal{S}_{C_{\mathrm{VO}}} \times \mathcal{S}_{C_{\mathrm{VI}}}, \end{aligned} \tag{1}$$

where the set $\mathcal{S}_{\mathrm{AP}k}$ for $k = 1, 2$ denotes the set of the state of AP k. The set $\mathcal{S}_x$ denotes the set of the APs at which the VO packets arrived. The set $\mathcal{S}_A$ denotes the set of the number of VO packets having arrived in the past. The sets $\mathcal{S}_{Q_{\mathrm{VO}}}$ and $\mathcal{S}_{Q_{\mathrm{VI}}}$ denote the set of the number of packets waiting in the VO queue and in the VI queue, respectively. The sets $\mathcal{S}_{C_{\mathrm{VO}}}$ and $\mathcal{S}_{C_{\mathrm{VI}}}$ denote the set of the backoff count of AC_VO and

AC_VI, respectively. The sets $\mathcal{S}_x, \mathcal{S}_A, \mathcal{S}_{Q_{\rm VO}}, \mathcal{S}_{Q_{\rm VI}}, \mathcal{S}_{C_{\rm VO}}$, and $\mathcal{S}_{C_{\rm VI}}$ are expressed as follows:

$$\begin{aligned}\mathcal{S}_x &= \{1, 2\},\\ \mathcal{S}_A &= \{0, 1, \ldots, N-1\},\\ \mathcal{S}_{Q_{\rm VO}} &= \{0, 1, \ldots, Q_{\rm VOmax}\},\\ \mathcal{S}_{Q_{\rm VI}} &= \{0, 1, \ldots, Q_{\rm VImax}\},\\ \mathcal{S}_{C_{\rm VO}} &= \{0, 1, \ldots, CW_{\max}[\text{AC\_VO}]\},\\ \mathcal{S}_{C_{\rm VI}} &= \{0, 1, \ldots, CW_{\max}[\text{AC\_VI}]\}.\end{aligned} \tag{2}$$

In (2), $Q_{\rm VOmax}$ and $Q_{\rm VImax}$ denote the maximum number of packets in the VO queue and the VI queue, respectively. Moreover, $CW_{\max}[\text{AC\_VO}]$ and $CW_{\max}[\text{AC\_VI}]$ denote the maximum value of the contention window (CW) of AC_VO and AC_VI, respectively. The state is observed when a VO packet arrives at an AP. The state $s_n \in \mathcal{S}$ for $n = 1, 2, \ldots, 2N$ denotes the state when the nth VO packet, which arrives after $n-1$ VO packets arrived in the episode, arrives at an AP.

The action is mapping each packet that arrives at an AP to AC_VO or AC_VI. We define the action set as follows:

$$\mathcal{A} = \mathcal{A}_1 \cup \mathcal{A}_2, \tag{3}$$

where $\mathcal{A}_k$ for $k = 1, 2$ denotes that the central controller maps the packet that arrives at an AP k to AC_VO and AC_VI, respectively. $\mathcal{A}_k$ for $k = 1, 2$ is expressed as follows:

$$\mathcal{A}_k \coloneqq \{\text{VO}_k, \text{VI}_k\}, \tag{4}$$

where $a_n \in \mathcal{A}$ for $n = 1, 2, \ldots, 2N$ denotes the action when the nth VO packet arrives at an AP.

We define the reward as the sign inversion of the transmission delay. The transmission delay depends on an overall sequence of states and actions. That is, the transmission delay is expressed as $t_N(\tau)$ where $\tau \coloneqq (s_1, a_1, s_2, a_2, \ldots, s_{2N}, a_{2N})$ is an overall sequence of states and actions. We set the reward as $r_N(\tau)$, which is given as $r_N(\tau) = -t_N(\tau)$.

B. Update Equation in Policy Gradient RL

We obtained the update equation for the parameter $\boldsymbol{\theta}$ in this system. First, we set the objective function, and the objective function was parameterized by $\boldsymbol{\theta}$. The update equation is given using gradient descent. However, the update equation is difficult to calculate analytically. Even in such a case, the update equation can be obtained by Monte Carlo approximation [9]. Moreover, by introducing an optimal baseline, the accuracy of the Monte Carlo approximation can be increased [10]. Therefore, we obtained the update equation in this system by introducing an optimal baseline by Monte Carlo approximation.

Because we aimed to minimize the expected value of the transmission delay, we set the objective function as the sign inversion of the expected value of the reward, which is given as $\mathbb{E}_\tau[r(\tau)]$. Because the policy $\pi_{\boldsymbol{\theta}}(a \mid s)$ is parameterized by $\boldsymbol{\theta}$ in policy gradient RL, τ is also parameterized by $\boldsymbol{\theta}$. Therefore, the objective function $\mathbb{E}_\tau[r(\tau)]$ is parameterized by $\boldsymbol{\theta}$.

The objective function $\mathbb{E}_\tau[r(\tau)]$ is expressed as follows:

$$J(\boldsymbol{\theta}) = \mathbb{E}_\tau[r(\tau)]. \tag{5}$$

By using gradient descent, the update equation of the parameter $\boldsymbol{\theta}$ is given as follows:

$$\boldsymbol{\theta}^{(k+1)} = \boldsymbol{\theta}^{(k)} + \eta \nabla_{\boldsymbol{\theta}} J\left(\boldsymbol{\theta}^{(k)}\right). \tag{6}$$

where η is the learning rate. By using baseline $b \in \mathbb{R}$, the gradient of $J(\boldsymbol{\theta})$ is given as follows:

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \mathbb{E}_\tau\left[(r(\tau) - b)\left(\sum_{n=1}^{2N} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a_n \mid s_n)\right)\right]. \tag{7}$$

(7) are difficult to calculate analytically. Therefore, we obtained the update equation of the parameter $\boldsymbol{\theta}$ by Monte Carlo approximation. The variance of $(r(\tau) - b)\left(\sum_{n=1}^{2N} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a_n \mid s_n)\right)$ becomes 0 when the baseline b satisfies the following equation [9]:

$$b = \frac{\mathbb{E}_\tau\left[r(\tau)\left(\sum_{n=1}^{2N} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a \mid s)\right)^2\right]}{\mathbb{E}_\tau\left[\left(\sum_{n=1}^{2N} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a \mid s)\right)^2\right]}. \tag{8}$$

By Monte Carlo approximation, the baseline b is given as follows [10]:

$$\begin{aligned} b &= \frac{\sum_{m=1}^{M} r(\tau^m)\left(\sum_{n=1}^{2N} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a_n^m \mid s_n^m)\right)^2}{\sum_{m=1}^{M}\left(\sum_{n=1}^{2N} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a_n^m \mid s_n^m)\right)^2}\\ &= \frac{-\sum_{m=1}^{M} t_N(\tau^m)\left(\sum_{n=1}^{2N} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a_n^m \mid s_n^m)\right)^2}{\sum_{m=1}^{M}\left(\sum_{n=1}^{2N} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a_n^m \mid s_n^m)\right)^2}. \end{aligned} \tag{9}$$

In (9), $s_n^m \in \mathcal{S}$ and $a_n^m \in \mathcal{A}$ denote the state and action when the nth packet arrives in the mth episode, respectively. Moreover, $\tau^m = \{s_1^m, a_1^m, s_2^m, a_2^m, \ldots, s_{2N}^m, a_{2N}^m\}$ denotes the overall sequence of states and actions in the mth episode. In addition, $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$ is given by Monte Carlo approximation as follows:

$$\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) \approx -\frac{1}{M}\sum_{m=1}^{M}\Big(t_N(\tau^m) - b\Big)\sum_{n=1}^{2N} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a_n^m \mid s_n^m). \tag{10}$$

By substituting (10) into (6), the update equation is given as follows:

$$\begin{aligned}\boldsymbol{\theta}^{(k+1)} &= \boldsymbol{\theta}^{(k)}\\ &- \frac{\eta}{M}\sum_{m=1}^{M}\Big(t_N(\tau^m) - b\Big)\sum_{n=1}^{2N} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}^{(k)}}(a_n^m \mid s_n^m).\end{aligned} \tag{11}$$

C. Policy Design

We obtained $\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}^{(k)}}(a_n^m \,|\, s_n^m)$ in (11) to update $\boldsymbol{\theta}$ by defining the policy $\pi_{\boldsymbol{\theta}}(a \,|\, s)$. By using the softmax function, $\pi_{\boldsymbol{\theta}}(a \,|\, s)$ can be calculated [11]. The policy $\pi_{\boldsymbol{\theta}}(a \,|\, s)$ is given as follows:

$$\pi_{\boldsymbol{\theta}}(a \,|\, s) \coloneqq \frac{\exp\left(\boldsymbol{\theta}^{\mathrm{T}} \boldsymbol{\phi}(s,a)\right)}{\sum_{a' \in \mathcal{A}} \exp\left(\boldsymbol{\theta}^{\mathrm{T}} \boldsymbol{\phi}(s,a')\right)}, \tag{12}$$

where dimension number of the parameter $\boldsymbol{\theta}$ is equal to that of $\boldsymbol{\phi}(s,a)$, which is the feature vector and can be designed arbitrarily. From (12), $\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a \,|\, s)$ is given as follows:

$$\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a \,|\, s) = \boldsymbol{\phi}(s,a) - \sum_{a' \in \mathcal{A}} \pi_{\boldsymbol{\theta}}(a' \,|\, s) \boldsymbol{\phi}(s,a'). \tag{13}$$

Next, we designed the feature vector $\boldsymbol{\phi}(s,a)$ to calculate the policy and to update $\boldsymbol{\theta}$. Let a state s as $(S_0, S_1, \ldots, S_{10}) \in \mathcal{S}$, where an element of the state $S_0 \in \mathcal{S}_x$ denotes the AP to which a high priority packet arrives. Elements of the state $S_1, S_6 \in \mathcal{S}_A$ denote the number of high priority packets having arrived in the past at AP 1 and AP 2, respectively. Elements of the state $S_2, S_7 \in \mathcal{S}_{Q_{\mathrm{VO}}}$ denote the number of packets waiting in the VO queue of AP 1 and AP 2, respectively. Elements of the state $S_3, S_8 \in \mathcal{S}_{Q_{\mathrm{VI}}}$ denote the number of packets waiting in the VI queue of AP 1 and AP 2, respectively. Elements of the state $S_4, S_9 \in \mathcal{S}_{C_{\mathrm{VO}}}$ denote the backoff count of the VO queue of AP 1 and AP 2, respectively. Elements of the state $S_5, S_{10} \in \mathcal{S}_{C_{\mathrm{VI}}}$ denote the backoff count of the VI queue of AP 1 and AP 2, respectively. Also, let an action a be $a \in \mathcal{A}_{S_0}$. However, we designed $\boldsymbol{\phi}(s,a)$ as a function of a, S_0 at first because a, S_0 and $S_1, \ldots, S_{10}$ have different characteristics. Elements of the state $S_1, \ldots, S_{10} \in \mathbb{N} \cup \{0\}$ denote numbers of such as packets waiting in queue or the backoff count. On the other hand, $a \in \mathcal{A}_{S_0}, S_0 \in \{1,2\}$ denote the index of AP.

First, because the value of $\boldsymbol{\phi}(s,a)$ changes as a and S_0 change, we set $\boldsymbol{\phi}_{\alpha,\chi}(s,a) \in \mathbb{R}^d$ as a function equal to $\mathbf{0}$ when a and S_0 are not equal to $\alpha \in \mathcal{A}_{S_0}$ and $\chi \in \mathcal{S}_x$, respectively. Using $\boldsymbol{\varphi}(S_1, S_2, \ldots, S_{10})$, we set $\boldsymbol{\phi}_{\alpha,\chi}(s,a)$ as follows:

$$\boldsymbol{\phi}_{\alpha,\chi}(s,a) = \boldsymbol{\varphi}(S_1, S_2, \ldots, S_{10}) \mathbb{1}(S_0 = \chi \wedge a = \alpha), \tag{14}$$

where $\mathbb{1}(\cdot)$ is an indicator function. By using $\boldsymbol{\theta}_{\alpha,\chi} \in \mathbb{R}^d$ and $\boldsymbol{\phi}_{\alpha,\chi}(s,a) \in \mathbb{R}^d$, $\boldsymbol{\theta} \in \mathbb{R}^{4d}$ and $\boldsymbol{\phi}(s,a) \in \mathbb{R}^{4d}$ are expressed as follows:

$$\boldsymbol{\theta} = \begin{bmatrix} \boldsymbol{\theta}_{\mathrm{VO}_{S_0},1} \\ \boldsymbol{\theta}_{\mathrm{VO}_{S_0},2} \\ \boldsymbol{\theta}_{\mathrm{VI}_{S_0},1} \\ \boldsymbol{\theta}_{\mathrm{VI}_{S_0},2} \end{bmatrix}, \boldsymbol{\phi}(s,a) = \begin{bmatrix} \boldsymbol{\phi}_{\mathrm{VO}_{S_0},1}(s,a) \\ \boldsymbol{\phi}_{\mathrm{VO}_{S_0},2}(s,a) \\ \boldsymbol{\phi}_{\mathrm{VI}_{S_0},1}(s,a) \\ \boldsymbol{\phi}_{\mathrm{VI}_{S_0},2}(s,a) \end{bmatrix}, \tag{15}$$

where $d \in \mathbb{R}$ denotes the degree of $\boldsymbol{\theta}_{\alpha,\chi}$ and $\boldsymbol{\phi}_{\alpha,\chi}(s,a)$.

Next, because the value of $\boldsymbol{\varphi}(S_1, S_2, \ldots, S_{10})$ changes as $S_1, S_2 \ldots, S_{10}$ change, we set $\boldsymbol{\theta}_{\alpha,\chi}^{\mathrm{T}} \boldsymbol{\phi}_{\alpha,\chi}(s,a)$ as a Dth order polynomial when S_0 and a are equal to χ and α, respectively.

TABLE I
SIMULATION PARAMETERS

Number of trials to obtain transmission delay	1000
Number of updates $\boldsymbol{\theta}$ K	100
Number of sending VO packets N	10
Number of episodes M	$1,000$
Learning rate η	10^{-4}
Arrival rate of VO packets λ_{VO}	$5 \times 10^5\,\mathrm{s}^{-1}$
Arrival rate of VI packets λ_{VI}	$2.5 \times 10^5\,\mathrm{s}^{-1}$
Slot time σ_s	$9\,\mu$s
Transmission time of the data frame t_{DATA}	$248\,\mu$s
Transmission time of the ACK frame t_{ACK}	$24\,\mu$s
AIFS of AC_VO and AC_VI $AIFS[\mathrm{AC}]$	$16\,\mu$s
Degree in (16) D	2
Constant in (18) γ	$1/5$
Constant in (18) δ	1

TABLE II
DEFAULT PARAMETERS OF THE EDCA SCHEME

	$CW_{\min}$	$CW_{\max}$	$AIFSN$
AC_VO	3	7	2
AC_VI	7	15	2

$\boldsymbol{\varphi}(S_1, S_2, \ldots, S_{10})$ is expressed as follows:

$$\boldsymbol{\varphi}(S_1, S_2, \ldots, S_{10}) = \begin{bmatrix} \varphi_{0,0,\ldots,0}(S_1, S_2, \ldots, S_{10}) \\ \varphi_{1,0,\ldots,0}(S_1, S_2, \ldots, S_{10}) \\ \vdots \\ \varphi_{0,0,\ldots,D}(S_1, S_2, \ldots, S_{10}) \end{bmatrix}. \tag{16}$$

In (16), $\varphi_{\sigma_1,\ldots,\sigma_{10}}(S_1, S_2, \ldots, S_{10}) \in \mathbb{R}$ is given as follows:

$$\varphi_{\sigma_1,\ldots,\sigma_{10}}(S_1, S_2, \ldots, S_{10}) = \prod_{k=1}^{10} \left(S_k'\right)^{\sigma_k}. \tag{17}$$

In (17), the variable S_k' for $k = 1, 2, \ldots 10$ denote variable transformation of S_k for learning the policy to enhance the reward, which is expressed as follows:

$$S_k' = \gamma_k (S_k + \delta_k), \tag{18}$$

where γ_k and δ_k are constants.

D. Procedure

The procedure of policy gradient RL is as follows.

1) Initialize $\boldsymbol{\theta}^{(0)} = \mathbf{0}$.
2) Calculate $\pi_{\boldsymbol{\theta}}(a \,|\, s)$ according to (12).
3) Simulate using $\pi_{\boldsymbol{\theta}}(a \,|\, s)$ and obtain $t(\tau^m)$, $\varpi_{\boldsymbol{\theta}}(\tau)$.
4) Calculate b according to (9).
5) Calculate $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$ according to (10).
6) Update $\boldsymbol{\theta}$ according to (6).
7) Repeat form 2) to 6) K times.

In this procedure, let $\varpi_{\boldsymbol{\theta}}(\tau)$ be $\sum_{n=1}^{2N} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a_n \,|\, s_n)$.

IV. SIMULATION

We obtained the optimal mapping algorithm using policy gradient RL in this system. In this system, APs transmit packets based on 802.11a [12]. The simulation parameters are shown in Table I.

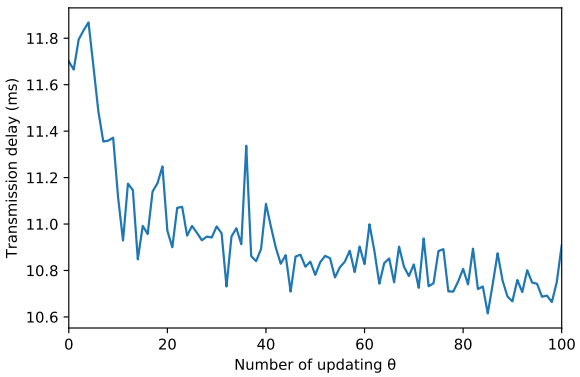

Fig. 3. Learning curve of the proposed mapping algorithm. The transmission delay was reduced mostly as updating $\boldsymbol{\theta}$.

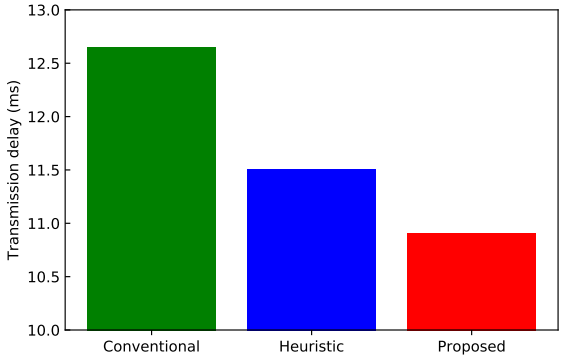

Fig. 4. Transmission delay of the proposed mapping algorithm and compared mapping algorithms. The transmission delay of the proposed mapping algorithm was shorter than those of the two compared mapping algorithms.

A. Policies of Compared Mapping Algorithms

We compared two mapping algorithms. Because the conventional EDCA mapping algorithm maps VO packets to AC_VO, the policy $\pi^{(1)}(a \mid s)$ used in the conventional EDCA algorithm is given as follows:

$$\pi^{(1)}(a \mid s) = \mathbb{1}(a = \mathrm{VO}_{S_0}). \tag{19}$$

We also considered a heuristic mapping algorithm based on the number of packets waiting in the queues. The heuristic mapping algorithm maps VO packets to AC with a smaller number of waiting packets. When the number of packets of AC_VO is equal to that of AC_VI, it maps to AC_VO which has more opportunities to transmit packets than does AC_VI. The policy $\pi^{(2)}(a \mid s)$ used in the heuristic mapping algorithm is expressed as follows:

$$\pi^{(2)}(a \mid s) = \begin{cases} \mathbb{1}(Q_{\mathrm{VO}} \leq Q_{\mathrm{VI}}), & a = \mathrm{VO}_{S_0}; \\ \mathbb{1}(Q_{\mathrm{VO}} > Q_{\mathrm{VI}}), & a = \mathrm{VI}_{S_0}. \end{cases} \tag{20}$$

B. Results from Simulation

Fig. 3 confirms that the transmission delay of the proposed mapping algorithm was reduced mostly as updating $\boldsymbol{\theta}$. Fig. 3 shows the learning curve of the proposed mapping algorithm, which is the relation between the number of updatings θ and the transmission delay.

Fig. 4 confirms that the transmission delay of the proposed mapping algorithm was smaller than that of two compared mapping algorithms. Fig. 4 shows the transmission delay of the proposed mapping algorithm and the compared mapping algorithms. The proposed mapping algorithm reduced the transmission delay by 13.8% compared to the conventional EDCA mapping algorithm. Furthermore, the proposed mapping algorithm reduced the transmission delay by 5.2% compared to the heuristic mapping algorithm.

Fig. 5 confirms that the number of packets waiting in the VO queue is less important than the number of packets waiting in the VI queue. Fig. 5 shows the relation between the probability of mapping VO packets that arrived at AP 1 to AC_VO $\pi_{\boldsymbol{\theta}}(\mathrm{VO}_{\mathrm{AP1}} \mid s)$, the number of packets waiting in the VO queue S_2, and the number of packets waiting in the VI queue S_3, where $S_i = 3$ for $i = 1, 4, 5, \ldots, 10$. We obtained the mapping algorithm based on the number of packets waiting in the VO queue. However, the probability $\pi_{\boldsymbol{\theta}}(\mathrm{VO}_{\mathrm{AP1}} \mid s)$ changed only a little as the number of packets waiting in the VO queue changed. On the other hand, the probability $\pi_{\boldsymbol{\theta}}(\mathrm{VO}_{\mathrm{AP1}} \mid s)$ changed greatly as the number of packets waiting in the VI queue changed. Therefore, although the mapping algorithm is based on the number of packets waiting in the high priority queue in previous research, the number of packets waiting in the VO queue is less important than the number of packets waiting in the VI queue.

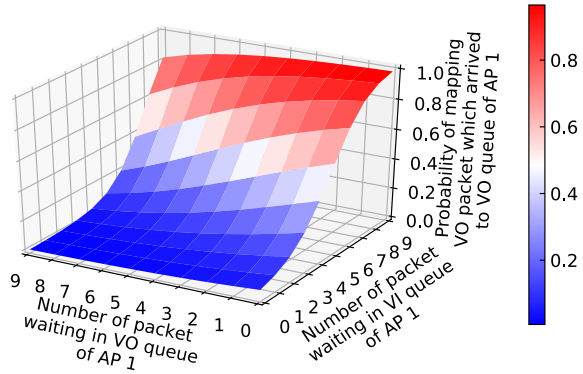

Fig. 5. Relation between the probability of mapping VO packets that arrived at AP 1 to AC_VO $\pi_{\boldsymbol{\theta}}(\mathrm{VO}_{\mathrm{AP1}} \mid s)$, the number of packets waiting in the VO queue S_2, and the number of packets waiting in the VI queue S_3. The probability of mapping VO packets to an AC decreases as the number of packets waiting in the queue of the AC increases. The number of packets waiting in the high priority queue, upon which the mapping algorithm in previous research was based, was less important than the number of packets waiting in the low priority queue.

Fig. 6 confirms that the transmission delay was reduced by mapping high priority packets from a long-term perspective. Fig. 6 shows the relation between the probability of

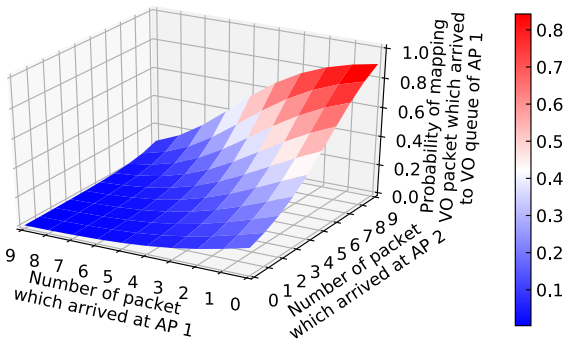

Fig. 6. Relation between the probability of mapping VO packets that arrived at AP 1 to the VO queue $\pi_{\boldsymbol{\theta}}(\mathrm{VO}_{\mathrm{AP1}} \,|\, s)$, the number of packets having arrived at AP 1 in the past S_1, and the number of packets having arrived at AP 2 in the past S_6. The proposed mapping algorithm maps high priority packets from a long-term perspective. By having an AP that takes a short time to complete transmission give up some transmission opportunities, an AP that takes longer to complete transmission obtains transmission opportunities.

mapping VO packets that arrived at AP 1 to the VO queue $\pi_{\boldsymbol{\theta}}(\mathrm{VO}_{\mathrm{AP1}} \,|\, s)$, the number of packets having arrived at AP 1 in the past S_1, and the number of packets having arrived at AP 2 in the past S_6, where $S_i = 3$ for $i = 2, 3, 4, 7, 8, 9, 10$. The probability $\pi_{\boldsymbol{\theta}}(\mathrm{VO}_{\mathrm{AP1}} \,|\, s)$ decreased as the number of packets having arrived at AP 1 in the past increased and as the number of packets having arrived at AP 2 in the past decreased. An AP with a small number of packets having arrived is considered to take a longer time to complete transmission of 10 VO packets compared to an AP with a large number of packets having arrived. Therefore, by having an AP with a large number of arriving packets give up transmission opportunities so that an AP with a smaller number of arriving packets could obtain transmission opportunities, the transmission delay was decreased.

V. Conclusions

We proposed using policy gradient RL to obtain a new mapping algorithm optimal for minimizing the transmission delay. We designed states that involved the number of packets having arrived in the past, actions, and rewards in RL. To solve the RL task, we used policy gradient RL and obtained the update equation of policy gradient RL in this system. For policy gradient RL, we designed a feature vector suitable for learning the policy to enhance the reward. The simulation results show that the proposed mapping algorithm reduces the transmission delay in relation to the mapping algorithms to which it was compared. Furthermore, the proposed mapping algorithm maps high priority packets from a long-term perspective.

Acknowledgement

This work was supported in part by JSPS KAKENHI Grant Number JP18H01442.

References

[1] "Wireless LAN medium access control (MAC) and physical layer (PHY) specifications: Medium access control (MAC) enhancements for quality of service (QoS)," IEEE Std. 802.11e-2007.

[2] C. H. Foh, Y. Zhang, Z. Ni, J. Cai, and K. N. Ngan, "Optimized cross-layer design for scalable video transmission over the IEEE 802.11 e networks," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 17, no. 12, pp. 1665–1678, Dec. 2007.

[3] R. Sadek, A. Youssif, and A. Elaraby, "MPEG-4 video transmission over IEEE 802.11e wireless mesh networks using dynamic-cross-layer approach," *Natl. Acad. Sci. Lett.*, vol. 38, no. 2, pp. 113–119, Feb. 2015.

[4] C. Lin, C. Shieh, C. Ke, N. K. Chilamkurti, and S. Zeadally, "An adaptive cross-layer mapping algorithm for MPEG-4 video transmission over IEEE 802.11e WLAN," *TeleCommun. Syst.*, vol. 42, no. 3–4, pp. 223–234, July 2009.

[5] X.-W. Yao, W.-L. Wang, S.-H. Yang, Y.-F. Cen, X.-M. Yao, and T.-Q. Pan, "IBP-frame adaptive mapping mechanism for video transmission over IEEE 802.11e WLANs," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 2, pp. 5–12, Apr. 2014.

[6] M. A. Labiod, M. Gharbi, F.-X. Coudoux, P. Corlay, and N. Doghmane, "Cross-layer scheme for low latency multiple description video streaming over Vehicular Ad-hoc NETworks (VANETs)," *AEU-International Journal of Electronics and Communications*, vol. 104, pp. 23–34, Feb. 2019.

[7] R. S. Sutton and A.G.Barto, *Reinforcement Learning: An Introduction*. MIT, 1998.

[8] J. Peters and S. Schaal, "Policy gradient methods for robotics," in *Proc. IROS 2006*, Beijing, China, Oct. 2006, pp. 2219–2225.

[9] ——, "Reinforcement learning of motor skills with policy gradients," *Neural netw.*, vol. 21, no. 4, pp. 682–697, Feb. 2008.

[10] M. P. Deisenroth, G. Neumann, J. Peters *et al.*, "A survey on policy search for robotics," *Found. Trends Robot*, vol. 2, no. 1–2, pp. 1–142, Aug. 2013.

[11] R. S. Sutton, D. A. McAllester, S. P. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Proc. NIPS 2000*, Denver, CO, USA, Nov. 2000, pp. 1057–1063.

[12] "Wireless LAN medium access control (MAC) and physical layer (PHY) specification: high-speed physical layer in the 5 GHz band," IEEE Std. 802.11a-2000.