ns3-ai: Rate Control for Wireless LAN by Deep Q-Network

Tomoki Nakashima*, Leonardo Lanante†, Muhammad Harry Bintang Pratama*, Masayuki Kurosaki*, and Hiroshi Ochi*
*Graduate School of Computer Science and Systems Engineering, Kyushu Institute of Technology, Japan
†Ofinno LLC, USA

***Abstract*—Transmission rate control in wireless LANs is one of the factors that affect communication quality. Many transmission rate control algorithms have been proposed in previous studies. However, there are cases where existing algorithms cannot adaptively control the rate due to the dynamics of wireless communication. In this paper, we propose a transmission rate control method based on Deep Q-Network (DQN), in which a DQN agent learns information about the communication environment and adaptively controls the transmission rate in response to the communication environment. We evaluate the proposed DQN-based transmission rate control by using the ns3-ai framework and the ns-3 network simulator. Simulations show that the proposed method improves throughput by up to 95% compared to the Minstrel existing method.**

I. Introduction

Currently, many electronic devices, such as smartphones, tablets, home appliances, and drones communicate wirelessly. It is expected that everything will be connected to wireless in the future [1]. The IEEE 802.11 standard network called Wi-Fi is still one of the most common wireless networks, with more than 900 million units in use [2]. In addition, with new broadband services offered, such as 4K/8K video streaming and XR (Extended Reality) technology, Wi-Fi requires high throughput and low latency. Wi-Fi networks have difficult channel predictability due to wireless medium sharing, node mobility, channel fading, and interference [4]. These channel dynamics degrade the wireless communication quality.

The transmission rate is a network parameter that determines how fast a node transmits data to the wireless medium. In a good communication environment (e.g., short distance and no interference), a higher transmission rate results in higher goodput and shorter channel occupancy time. On the other hand, in a poor communication environment, such as a dense environment, selecting a high transmission rate increases the probability of packet loss and degrades communication quality. Therefore, appropriate transmission rate control is necessary to provide communications that stably satisfy high transmission rate requirements.

Adaptive control of the transmission rate is necessary to maximize throughput under real-world conditions. Throughput can be maximized by selecting the transmission rate according to changes in the communication environment. Many transmission rate control algorithms have been proposed in previous studies [8][9][10]. However, these existing algorithms sometimes fail to cope with dynamic changes in the communication environment. Therefore, in a previous study, a method using Q-learning was proposed to control the transmission rate by adapting to dynamic communication environments [5]. The results in [5] show that the transmission rate control by Q-learning is superior to Minstrel, an existing method. In [5], transmission rate control using Deep Q-network, which can perform more complex processing, is introduced as a future study.

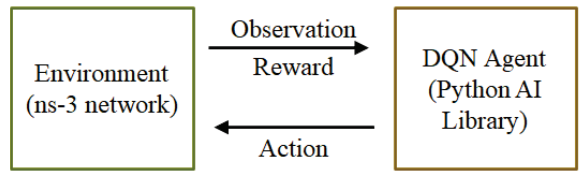

Fig. 1. ns3-ai overview illustration

We propose an adaptive transmission rate control method for wireless LANs(WLAN) using Deep Q-network (DQN) in dynamic environments where terminals are moving. We developed a model for adaptive transmission rate control using DQN and conducted simulation evaluations. The simulations were conducted using ns3-ai, a platform that combines the network simulator ns-3 with AI processing in Python[6]. The reinforcement learning agent can learn from the simulated network through observations and rewards, and the actions chosen by the agent for the following time steps can be applied to the simulated network, as illustrated in Fig.1.

The contributions of this research are as follows:

- We design a novel reinforcement learning agent with DQN for adaptive WLAN transmission rate control. The agent controls the data rate by changing the MCS.
- We show how the reinforcement learning agent with DQN works using ns3-ai. We also compare the results with Minstrel, an existing transmission rate control method that is well-known in general.

This paper is organized as follows. In Section 2, we describe the WLAN transmission rate and evaluate the performance of existing methods by simulation. The proposed transmission rate control method using DQN is described in Section 3. In Section 4, we evaluate the proposed method by simulation. Section 5 concludes the paper.

Copyright © 2021 IEICE. Permission request for reproduction: Service Department, IEICE Headquarters Office, E-mail: service@ieice.org. IEICE Provisions on Copyright: https://www.ieice.org/eng/copyright/.

II. Rate Control in Wireless LAN

IEEE 802.11 has a transmission rate index called the Modulation and Coding Scheme (MCS). In the IEEE 802.11 WLAN standards, different MCS can be selected for every data packet. Table I shows the IEEE 802.11ac MCS and the corresponding physical data rates of each MCS, with a bandwidth of 20MHz and the number of spatial streams of 1. A specific Signal to Interference plus Noise Ratio (SINR) is required for correct signal reception at each MCS: the higher the MCS, the higher the required SINR, and conversely, the lower the required SINR at the lower MCS. MCS selection, which will be referred to as rate control in this paper, plays an important role to improve the quality of communication. However, its implementation is outside the scope of the standard. Many rate control algorithms have been proposed in [8][9][10]. The most popular algorithm is Minstrel and its extension Minstrel-HT [11], which are already implemented in operating systems such as Linux.

Minstrel selects an MCS based on a sampling process. Minstrel collects statistics on transmission attempts for each MCS. The actual throughput is evaluated as the probability of successful transmission multiplied by the packet payload and divided by the packet transmission time. Before sending a new frame, Minstrel determines the following sequence:

1) the MCS with the highest throughput
2) the MCS with the second-highest throughput
3) the MCS with the highest probability of successful transmission
4) the lowest MCS

The maximum number of consecutive transmission attempts for each of these MCS is also defined. This is to limit the duration of the attempts. In addition, by default, the algorithm operates in "look-around" mode with a 10% and tries a random MCS. This replaces the MCS with the highest throughput or the second-highest throughput.

We performed a performance evaluation simulation of Minstrel in ns-3. The parameters of the simulation are shown in Table II. In the log-distance path loss model, the propagation loss PL at a certain distance d is expressed as

$$PL = L_0 + 10n \log_{10} \frac{d}{d_{ref}} \tag{1}$$

where the propagation losses (L_0 and propagation loss at the reference distance PL) are in dBm, n is the path loss exponent, and the distances (the reference distance d_{ref} and d) are in meter. L_0 should not be dB, but dBm, because it is the power loss.

In this paper, we assume that $L_0 = 50$dBm, $n = 3.5$, $d_{ref} = 1$m. To further simplify the simulation, we also assume a full buffer DL traffic model with packets always ready to be transmitted. The number of episodes per simulation is the interval at which the throughput is measured and is adjusted to a value that makes it easy to understand the performance. We consider the following simulation scenario: two terminals, each acting as a transmitter and a receiver, respectively, communicate while the receiving terminal is moving away at the speed of 7m/s. The simulation uses fixed MCS 0-8 or Minstrel rate control. The average MAC throughput at 0.06s intervals is shown in Fig. 2. It shows that Minstrel causes a sharp drop in throughput when the appropriate MCS is switched. There are also other areas where throughput drops. This is due to the random selection of MCSs by the "look around" mode.

TABLE I
Modulation and Coding Scheme in IEEE 802.11ac

MCS Index	Spatial streams	Modulation	Coding	Data rate[Mbps] 20MHz 0.8µs GI	0.4µs GI
0	1	BPSK	1/2	6.5	7.2
1	1	QPSK	1/2	13	14.4
2	1	QPSK	3/4	19.5	21.7
3	1	16-QAM	1/2	26	28.9
4	1	16-QAM	3/4	39	43.3
5	1	64-QAM	2/3	52	57.8
6	1	64-QAM	3/4	58.5	65
7	1	64-QAM	5/6	65	72.2
8	1	256-QAM	3/4	78	86.7

TABLE II
Simulation parameters

Standard	IEEE 802.11ac
Propagation Model	Log-distance propagation
Noise Floor	-94dBm
RTS/CTS	Disable
Bandwidth	20MHz
Guard Interval	0.8µs
Channel number	36(5180MHz)
Transmit power	20dBm
Number of AP/STA	1/1
Number of Antennas	1
Traffic type	UDP, DL, Full Buffer
Carrier sense threshold	-100dBm
Simulation duration	10s
Number of Steps per sim	180
Step time	0.06s

III. Rate Control by Deep Q-Network

Deep Q-network (DQN), also known as deep reinforcement learning, is a learning method that repeatedly tries to approach the correct answer from multidimensional information [12]. By using a neural network, it is possible to learn complex environments with multidimensional input. This section describes the design and algorithm of the proposed DQN rate control.

A. System Overview

Fig. 3 shows a system overview. Throughput, current MCS, and distance are simulated in the ns-3. The throughput is the amount of data received per simulation step divided by the step time. It is used to calculate the reward, which will be described in the next subsection, DQN Framework. The neural network is updated with the reward calculated based on the throughput. Current MCS and The distance between the transmitter and receiver are used as input to the neural network. Actions are selected by ϵ-greedy between exploitative action by the neural

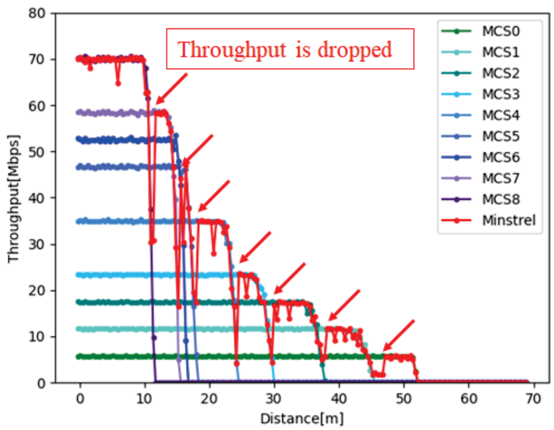

Fig. 2. MAC throughput for mobility scenarios from 0m to 70m

TABLE III
DISCRETIZATION OF DISTANCE

Distance[m]	Discretized (one-hot)
43.77∼	[100000000]
36.33∼43.77	[010000000]
28.24∼36.33	[001000000]
23.07∼28.24	[000100000]
16.77∼23.07	[000010000]
15.45∼16.77	[000001000]
14.32∼15.45	[000000100]
10.46∼14.32	[000000010]
∼10.46	[000000001]

network and random exploratory action. As shown in Fig. 4, the neural network architecture is as follows.

- Input layer: 18 units of discretized MCS and distance
- Intermediate layer: 126 units (ReLu functions)
- Output layer: 3 units of up one MCS, not changing, or down one MCS (Linear function)

The discretization of the distance is shown in Table III. The distances in Table III were calculated by the log-distance path loss model in equation (1) based on the SINR required to receive signal in Table IV. The SINRs in Table IV were calculated experimentally using ns-3 simulations. Exploitative action was chosen to achieve the maximum value of the output of the neural network, while the exploratory action was chosen to give a random output. The actions are determined by ϵ-greedy, which is updated by a specific rule. Once an action has been determined by ϵ-greedy, it is passed to ns-3 for simulation in the next time step.

B. DQN Framework

We describe in more detail the learning algorithm of DQN, which is to maximize the expected long-term reward. The reward is obtained from the environment after an action has been taken. In this paper, the reward is -100 when the throughput is 0Mbps. Otherwise, the reward is the difference between the throughput at current and the previous time step,

TABLE IV
THE SINR REQUIRED TO RECEIVE SIGNAL

MCS Index	S_0[dB]	Discretized MCS (one-hot)
0	3.97	[100000000]
1	6.55	[010000000]
2	9.39	[001000000]
3	13.21	[000100000]
4	16.29	[000010000]
5	21.13	[000001000]
6	22.38	[000000100]
7	23.54	[000000010]
8	28.31	[000000001]

as shown in the following equation

$$r_t = \begin{cases} -100, & Tp_{t+1} = 0. \\ Tp_{t+1} - Tp_t, & \text{otherwise.} \end{cases} \quad (2)$$

where Tp is throughput in Mbps.

The agent determines its action a from the output of the Q-network based on the state s obtained from the environment, as shown in the following equation.

$$a_t = \arg\max\, Q(s_t, a_t) \quad (3)$$

In this study, action a is a set of three options: increase one MCS, keep the current MCS, or decrease one MCS. As has been mentioned, ϵ-greedy is used as the exploration policy. To allow for uniform search, if action is chosen to move down the MCS at MCS 0, it transitions to MCS 8. Similarly, if action is chosen to move up the MCS at MCS 8, it transitions to MCS 0.

When an agent acquires a new reward through action, the $Q(s_t, a_t)$ is updated with the following rules for the state and action of the previous time step.

$$\begin{aligned} &Q(s_t, a_t) \leftarrow Q(s_t, a_t) \\ &+\alpha[r_{n+1} + \gamma \max\, Q(s_{t+1}, a_{t+1}) - Q(s_t, a_t)] \end{aligned} \quad (4)$$

α indicates the learning rate at (0,1]. γ is also (0,1] and indicates the discount rate. The discount rate is a value that indicates what proportion of the long-term future reward is used for the current update. In DQN, a neural network is trained to update $Q(s_t, a_t)$. It can be trained by minimizing the loss function of the neural network. The loss function is given by:

$$E = [r_{n+1} + \gamma \max\, Q(s_{n+1}, a_{n+1}) - Q(s_t, a_t)]^2 \quad (5)$$

In this paper, the neural network is updated using the optimization function Adam.

IV. NS-3 SIMULATION

A. Stationery Scenario

First, a simulation evaluation at a fixed location was performed to show that the designed DQN rate control works. In this scenario, a transmitter (Tx) and a receiver (Rx) are 10m away from each other; the two nodes are quite close to each other, and there are no other interfering nodes. Therefore, bandwidth can be used efficiently by using higher MCS levels.

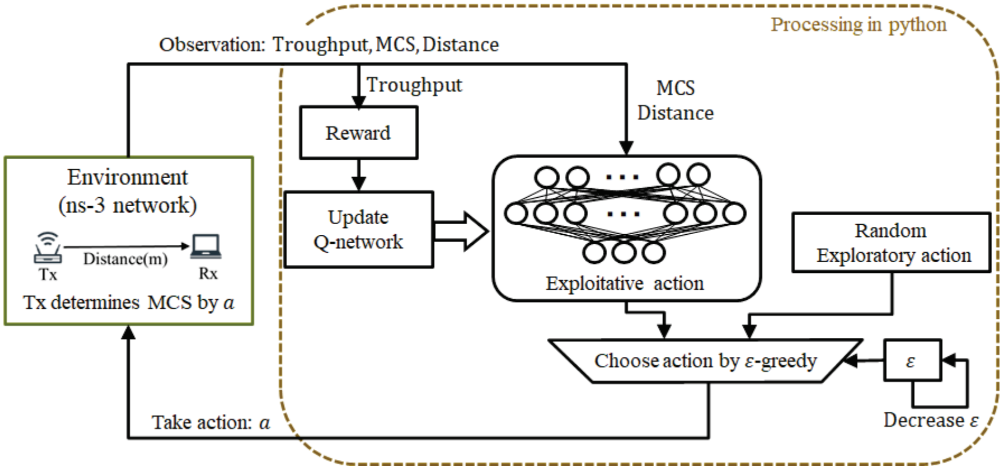

Fig. 3. Illustration of System Overview

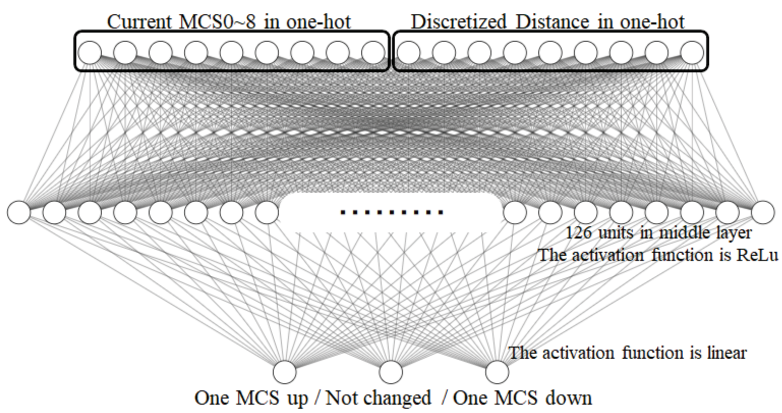

Fig. 4. Illustration of Q-network

Table V shows the parameters for learning and the changes made from the simulation parameters in Table II in Section 2. The transmission rate of data packets is controlled by the DQN agent by increasing or decreasing the MCS level at each time step.

The change in the MCS selected by the DQN agent in 500 steps over 50 seconds is shown in Fig.5. The simulation results show that the DQN agent is able to select a higher MCS at the end of the simulation. The reason why the values are not stable during the learning process is due to the change in MCS caused by the random selection by ϵ-greedy. Furthermore, the average throughput at each step is shown in Fig. 6. Since there are no interfering nodes, the throughput increases as a higher MCS is selected.

B. Mobile Scenario

Next, we evaluate the DQN rate control in the mobile scenario. For simplicity, we focus on the throughput drop at 11m in Fig. 2. In this scenario, there are one Tx and one Rx, but the Rx moves away from 4m to 14m at 1m/s. Table VI shows the parameters for training process and the changes made from the simulation parameters in Table II in Section 2. In the mobility scenario, the agent learns 1000 episodes, with one episode of movement from 4m to 14 m. ϵ reduction is performed at each episode.

After training 1000 episodes, the learned model was evaluated using a scenario in which the Rx moves from 0m to 13m

TABLE V
MODIFIED SIMULATION PARAMETERS AND TRAINING PARAMETERS IN STATIONERY SCENARIO

Simulation duration	50s
Number of Steps per sim	500
Step time	0.1s
γ	0.9
α	0.1
ϵ initial value	1
ϵ minimum value	0
ϵ reduction	0.002

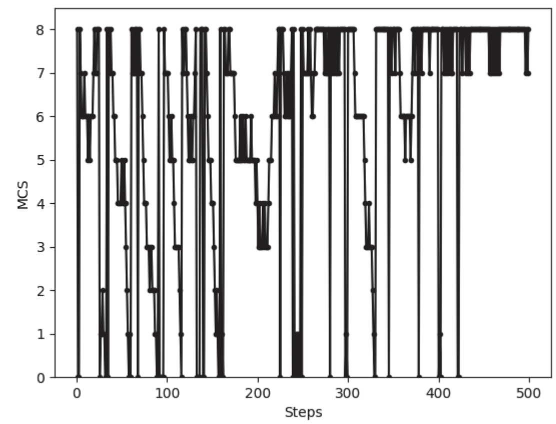

Fig. 5. MCS in the Stationery case(10m)

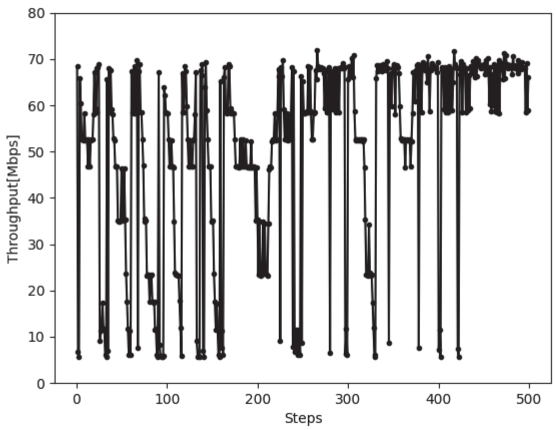

Fig. 6. Throughput in the Stationery case(10m)

at 7m/s. The results are shown in Fig. 7. The step interval was set to 0.6s, the same as in the simulation in Fig. 2. From Fig. 7, Minstrel shows a sharp drop in throughput at 11m when the optimal MCS switches from MCS 8 to MCS 7. However, the proposed DQN rate control seamlessly switches the MCS to the optimal one, preventing a sharp drop in throughput. Evaluation results of the proposed DQN rate control show that it improves throughput by up to 95%.

V. Conclusions

In this paper, we proposed a method for rate control using DQN with MCS and the distance between the transmitting and receiving nodes as inputs. ns3-ai was used to train a DQN agent to determine the suitable MCS based on the distance. We validated the proposed design in a stationery scenario where it is converged in the learning. The validation with a mobile scenario showed up to 95% improvement compared to Minstrel. Future work includes training DQN agents on longer moves and learning in more complex scenarios such as dense networks.

TABLE VI
Modified simulation parameters and training parameters in mobility scenario

Simulation duration	10s
Number of Step per sim	100
Step time	0.1s
Episode	1000
γ	0.9
α	0.1
ϵ	1
ϵ minimum value	0
ϵ reduction per episode	0.001

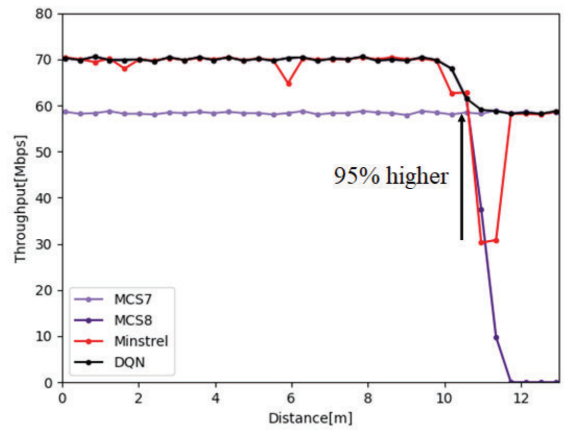

Fig. 7. Evaluation of throughput in the mobility case from 0m to 13m

References

[1] Cisco, Inc., *The internet of everything.* (2016). Accessed: June 9, 2022. [Online]. Available: https://www.cisco.com/c/dam/en_us/about/business-insights/docs/ioe-value-index-faq.pdf

[2] Wi-Fi Aliance. “Wi-Fi Alliance® publishes 2018 Wi-Fi® predictions.” Wi-Fi Alliance®. https://www.wi-fi.org/news-events/newsroom/wi-fi-alliance-publishes-2018-wi-fi-predictions (accessed June 9, 2022).

[3] Cisco, Inc. “IEEE 802.11ax: The Sixth Generation of Wi-Fi White Paper.“ Cisco.com https://www.cisco.com/c/en/us/products/collateral/wireless/white-paper-c11-740788.html (accessed June 9, 2022).

[4] T. Ngo “Why wi-fi stinks and how to fix it.“ IEEE Spectrum https://spectrum.ieee.org/why-wifi-stinksand-how-to-fix-it (accessed June 9, 2022).

[5] S. Cho, “Rate Adaptation with Q-Learning in CSMA/CA Wireless Networks,” *Journal of Information Processing Systems,* vol. 16, no.5, pp. 1048-1063, 2020.

[6] Y, Hao et al., “Ns3-Ai: Fostering Artificial Intelligence Algorithms for Networking Research,” *Proceedings of the 2020 Workshop on Ns-3,* WNS3 2020, pp. 57–64, 2020.

[7] *IEEE Standard for Information technology – Telecommunications and information exchange between systems Local and metropolitan area networks – Specific requirements Part 11: WLAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications,* IEEE Std 802.11-2016, 2016.

[8] W. Yin, P. Hu, J. Indulska, M. Portmann, and Y. Mao, “MAC-layer rate control for 802.11 networks: A survey,” *Wireless Netw,* vol. 26, pp. 3793–3830, 2020.

[9] S. Biaz and S. Wu, “Rate adaptation algorithms for IEEE 802.11 networks: A survey and comparison,” in *Proc. IEEE Symp. Comput. Commun.,* Jul. 2008, pp. 130–136.

[10] M. A. Gawas and R. Tambi, “Data rate adaptation algorithms survey for IEEE 802.11 networks,“ in *Proc. Int. Conf. Current Trends Comput., Electr., Electron. Commun. (CTCEEC),* Sep. 2017, pp. 926–932.

[11] F. Fietkau and D. Smithies. (2010). *Minstrel HT: New Rate Control Module for 802.11 n.* [Online]. Available: https://lwn.net/Articles/376765/
[12] Mnih, Volodymyr et al., "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602,* 2013.