Reinforcement Learning-Based Distributed Channel Access for Delay Optimization

Zhenyu Chen, *Student Member, IEEE*, Xinghua Sun, *Member, IEEE*, Huadong Li, Chenyuan Feng, *Member, IEEE*, Xijun Wang, *Member, IEEE*, Qiaofeng Xue, Tony Q. S. Quek, *Fellow, IEEE*

***Abstract*—As new applications evolve rapidly, wireless networks increasingly require low-delay communication to significantly enhance the quality of user experience. In response, the evolution of the medium access control (MAC) layer has gained more attention, particularly through the application of reinforcement learning to optimize access strategies. In order to meet the low-delay requirements, we propose a reinforcement learning-based MAC protocol, named soft actor-critic multiple access (SAC-MA). To mitigate frequent collisions caused by the exploratory behavior, we propose a multiple waiting actions mechanism that allows stations to wait for multiple time slots. This mechanism enables the agent to develop a more flexible and intelligent access strategy, thereby effectively reducing delay. Additionally, we introduce an innovative formulation in which the head-of-line packet is treated as the agent, enabling more timely feedback and observations. We conduct extensive simulations to demonstrate that SAC-MA: 1) reduces delay by approximately 27.9% and 56.5% compared to the conventional MAC protocol with standard parameters under the collision and capture models, respectively; 2) adapts to environmental changes in dynamic scenarios; 3) coexists harmoniously with legacy stations and reduces the network delay in heterogeneous scenarios. Finally, we perform ablation studies to evaluate the effectiveness of the proposed mechanisms.**

***Index Terms*—Medium access control, Distributed channel access, Wireless network, Reinforcement learning, Delay optimization**

I. Introduction

NOWADAYS, wireless technologies like cellular networks and Wi-Fi have become ubiquitous in daily life. Their rapid evolution has enabled a host of emerging applications, such as virtual reality [1] and robotics [2]. These applications impose more stringent requirements on wireless networks to guarantee high communication quality. Among these requirements, low latency is especially critical, as delays in data transmission can severely degrade user experience and compromise safety or system performance.

Across the layers of communication networks, the medium access control (MAC) layer plays a crucial role in reducing latency by efficiently regulating access to the shared medium among multiple stations. As a fundamental design at the MAC layer, *distributed channel access* enables stations to access the channel independently and distributedly. This approach is widely adopted in wireless networks such as Wi-Fi, Ultra-Wideband, and ZigBee, owing to its advantages in simplicity of deployment and scalability. Nevertheless, the lack of coordination leads to the channel contention among stations. Collisions occur when multiple stations attempt to access the channel simultaneously, resulting in transmission failures. To mitigate collisions, a random access mechanism is employed, allowing stations to randomly determine their transmission timing. One well-known MAC protocol that utilizes this mechanism is carrier sense multiple access with collision avoidance (CSMA/CA), widely used in Wi-Fi networks. In the CSMA/CA protocol, each station selects a random backoff interval from the contention window and waits before initiating transmission.

The conventional random access mechanism suffers from relatively low MAC efficiency [3]. To improve MAC layer performance, reinforcement learning (RL) has emerged as a promising approach to solve the distributed channel access problem. In RL-based MAC protocols, the stations are modeled as agents to maximize the expected cumulative rewards, which reflect network performance metrics such as throughput and delay. Compared with the random access protocols, RL-based protocols learn the access strategy utilizing the data obtained through interactions with the wireless environment. This data-driven method empowers the potential of stations to achieve higher performance without redundant backoff. While several RL-based MAC protocols have achieved great performance in terms of throughput and fairness, little attention has been paid to delay optimization.

To fill this gap in the literature, our work focuses on designing RL-based protocols aimed at delay optimization. Specifically, we consider the *tail delay* metric, which has

The work of Xinghua Sun was supported in part by Shenzhen Science and Technology Program under Grant CJGJZD20240729142100001, and in part by Guangdong Basic and Applied Basic Research Foundation under Grant 2024A1515012015. The work of Tony Q. S. Quek was supported in part by the National Research Foundation, Singapore and Infocomm Media Development Authority under its Communications and Connectivity Bridging Funding Initiative. The work of Xijun Wang was supported by the National Natural Science Foundation of China under Grant 62271513. (*Corresponding author: Xinghua Sun and Tony Q. S. Quek.*)

Zhenyu Chen and Xinghua Sun are with the School of Electronics and Communication Engineering, Shenzhen Campus of Sun Yat-sen University, Shenzhen 518107, China (email: chenzhy89@mail2.sysu.edu.cn; sunxinghua@mail.sysu.edu.cn).

Huadong Li and Qiaofeng Xue are with Signalwing Corporation, Shenzhen 518048, China (email: huadong.li@signalwing.com; qiaofeng.xue@signalwing.com).

Chenyuan Feng is with EURECOM, Sophia Antipolis 06410, France (e-mail: Chenyuan.Feng@eurecom.fr).

Xijun Wang is with the School of Electronics and Information Technology, Sun Yat-sen University, Guangzhou, China (e-mail: wangxijun@mail.sysu.edu.cn).

T. Q. S. Quek is with the Singapore University of Technology and Design, Singapore 487372, and also with the Department of Electronic Engineering, Kyung Hee University, Yongin 17104, South Korea (e-mail: tonyquek@sutd.edu.sg).

gained increasing attention in the latest Wi-Fi standard, IEEE 802.11bn [4]. This standard emphasizes improving the 95th percentile of the latency distribution. The tail delay refers to the extremely high latency experienced by a small fraction of requests, which can have a significant negative impact on the quality of user experience [5]. The challenge for delay optimization arises from two aspects. First, delay is sensitive to collisions, which can lead to packet buffer backlogs and increased latency. Nevertheless, the exploratory behavior of agents often causes frequent collisions among stations. Second, the delay experienced by a station changes slowly over time, resulting in delayed feedback to the agents and adversely affecting their training process. In order to tackle these challenges, we propose a MAC protocol based on a RL algorithm soft actor-critic (SAC) and name it soft actor-critic multiple access (SAC-MA). Our contributions and simulation results are presented below:

- To tackle the first challenge, we propose multiple waiting actions mechanisms. This mechanism allows stations to choose different waiting durations, helping distribute transmission attempts over time and consequently reducing collision probability.
- To tackle the second challenge, we design the packet-based agent mechanism to maximize the cumulative reward during the lifetime of the packet, which can obtain more timely feedback and observations.
- We conduct extensive experiments to evaluate the performance of SAC-MA under both the collision model and capture model. Simulation results demonstrate that SAC-MA outperforms both the classic CSMA/CA and RL-based baseline in static scenarios and dynamic scenarios. Furthermore, SAC-MA coexists harmoniously with the legacy CSMA/CA stations and reduces the network delay. Ablation studies evaluate the effectiveness of the proposed mechanisms.

The rest of this paper is organized in the following. Section II presents the literature review on RL-based MAC protocols. Section III introduces the system model. The proposed mechanisms are presented in Section IV. The RL formulation and algorithms are detailed in Section V. Simulation results are presented in Section VI, and conclusions along with future work are discussed in Section VII.

II. Related Work

Previous works on distributed channel access have focused on employing reinforcement learning to improve random access mechanisms. Typically, based on the framework of the CSMA/CA protocol, an agent is trained to select the contention window size to optimize network performance [6]–[14]. Among these studies, [6]–[12] aimed to optimize network throughput or fairness performance, while [13], [14] concentrated on improving delay performance. In [13], authors enhanced CSMA/CA by adjusting the contention window and clear channel assessment threshold in downlink Wi-Fi networks. In [14], the authors combined federated learning with reinforcement learning and introduced a training pruning strategy along with a weight aggregation algorithm. Although these works improved network performance, the underlying random access mechanism inherently limits them to relatively low MAC efficiency [3].

To enhance MAC efficiency, researchers have increasingly focused on designing MAC protocols entirely based on reinforcement learning [3], [15]–[23]. In these works, each station employs an RL agent to decide whether to transmit in a given idle slot. Compared to conventional backoff mechanisms, this method learns more effective access strategies without redundant backoff delays, thereby improving network performance. The distributed channel access problem is typically modeled as a multi-agent task, where agents at individual stations collaborate to optimize various performance metrics. The work in [3] proposed a novel MAC protocol, QMIX-advanced Listen-Before-Talk (QLBT), which outperforms CSMA/CA and its theoretical performance bound in terms of throughput, delay, and jitter. Addressing the hidden terminal problem, [15] proposed a new MAC protocol that outperforms CSMA/CA across various performance metrics. In [16] and [17], RL-based MAC protocols were designed for heterogeneous networks with single and multiple channels, respectively. In [18], the authors designed a novel global state representation and action mechanism to achieve near-optimal throughput and fairness in massive access scenarios. In [19], [20], authors introduced multi-task learning into multi-agent reinforcement learning to enhance adaptability in dynamic wireless networks. In [21], federated learning was utilized in a RL-based MAC protocol to achieve fairness among users. The works in [22] and [23] focused on simultaneously achieving max-min fairness and maximizing network throughput. In [24], the authors proposed a heterogeneous multi-agent reinforcement learning framework, enabling seamless collaboration between stations using different RL algorithms. Similar to our work, [25] proposed a fully decentralized multi-agent reinforcement learning framework for random access network optimization. By sharing local rewards across devices, the method eliminates the need for centralized training and reduces communication overhead. However, it only considered a single type of traffic pattern and did not conduct a comprehensive delay analysis under different traffic scenarios. In addition to reinforcement learning, some studies [26]–[28] leveraged multi-armed bandit algorithms to optimize the distributed channel access mechanism.

As previously introduced, most existing studies have targeted network throughput and fairness, using metrics such as α-fairness [15], [16], the Jain fairness index [18]–[20], and max-min fairness [22], [23]. Although [3] and [15] evaluated delay performance in comparison to the CSMA/CA protocol, their evaluations were confined to saturated-traffic scenarios. To the best of our knowledge, no comprehensive study on RL-based MAC protocol has been conducted to optimize the delay performance in distributed channel access problems.

Unlike distributed channel access, the scheduling-based framework employs a centralized controller to manage access for all devices. Within this framework, several studies have focused on MAC layer design for delay optimization in cellular networks [29]–[33]. In [29], the authors focused on a delay-oriented scheduling problem in the fifth-generation

network and proposed a recurrent proximal policy optimization algorithm to compensate for the lack of channel and queue information. The proposed algorithm outperformed existing methods in terms of both tail delay and average delay. In [30], the authors proposed the dynamic transmission and delay optimization random access scheme to adjust the backoff indicator in the massive machine-type communications scenarios. In [31], authors proposed the Q+-learning algorithm to improve the delay performance of throughput-optimal scheduling algorithms. To minimize the packet delays and packet drop rates, [32] proposed a scheduling framework that can select different scheduling rules. Based on a knowledge-assisted RL algorithm, the authors in [33] designed a scheduling policy for delay-sensitive traffic. Although these works achieved significant gains in terms of delay performance, they are not applicable to distributed channel access scenarios due to the absence of a centralized controller. Moreover, scheduling methods require frequent exchange of control messages, which can consume channel resources and potentially exacerbate overall delay.

III. System Model

In this work, we consider a wireless network where N stations communicate with an access point (AP) via a shared channel. Stations employ carrier sensing to detect whether the channel is idle. We assume that all stations can sense each other's transmissions, meaning no hidden terminals exist in the network. Adopting the IEEE 802.11 distributed coordination function (DCF), each station is permitted to transmit after the channel is sensed idle for the DCF interframe space (DIFS) duration. Time is divided into discrete time slots, and each station is permitted to initiate transmission at the beginning of each time slot. It is assumed that stations access the channel in a distributed manner, meaning that each station independently determines its transmission timing. To conduct a comprehensive evaluation, we consider two distinct models to characterize the conditions for successful transmission:

- Collision model [34]: One packet can be successfully received if other stations do not transmit during the packet transmission.
- Capture model [35]: One packet can be successfully transmitted as long as its received signal-to-interference-plus-noise ratio (SINR) exceeds a certain capture threshold μ. For station i, the received power $P_{r,i}$ can be written as $P_{r,i} = P_{t,i} \cdot |g_i|^2 \cdot |h_i|^2$, where $P_{t,i}$ denotes the transmission power of station i, and g_i and h_i denote the large-scale and small-scale fading coefficients, respectively. The wireless channel is assumed to undergo block Rayleigh fading, i.e., $|h_i|^2 \sim \exp(1)$ and $|h_i|^2$ varies from packet to packet. Assume that each station performs power control to combat the large-scale fading, meaning that $P_{t,i} = \frac{P}{|g_i|^2}$. The mean received signal-to-noise ratio (SNR) can be written as $\rho = P/\sigma^2$, where σ^2 denotes noise power. As a result, the received SINR value μ_i can be written as

$$\mu_i = \frac{|h_i|^2}{\sum_{j \in \mathcal{C}_i} |h_j|^2 + \frac{1}{\rho}}, \tag{1}$$

where $\mathcal{C}_i$ is the set of stations that transmit concurrently with station i.

The collision model represents a worst-case scenario in which the packet cannot be successfully received as long as it overlaps with any other packet. In contrast, the capture model assumes that each station's packet is decoded independently, treating signals from other stations as background noise. As a result, multiple packets may be successfully decoded simultaneously. After successful transmission, the station receives an acknowledgment (ACK) after short interframe space (SIFS) from AP. In the case of a transmission failure, the station will retransmit the packet until the number of attempts exceeds the maximum retransmission limit, after which the packet is dropped. For simplicity, only uplink data transmissions from stations to the AP are considered here.

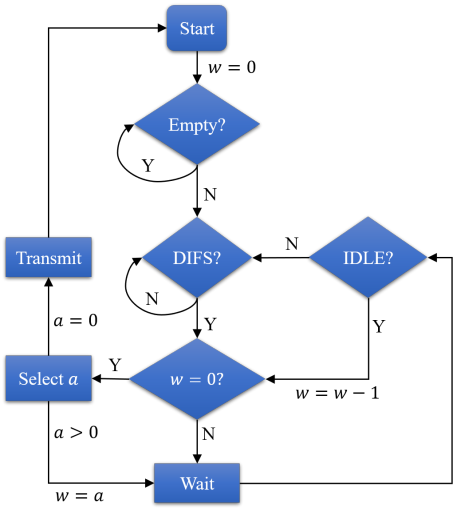


Fig. 1: The implementation process of the multiple waiting actions mechanism.

IV. Mechanism Design and Implementation

In this section, we present the design and implementation details of our proposed multiple waiting actions and packet-based agent mechanisms.

A. Multiple Waiting Actions Mechanism

In existing RL-based MAC protocols, the action space is typically defined as a binary decision between *transmitting a packet* and *waiting for one time slot*. This restricted action space often leads stations to transmit at inopportune moments due to exploration behaviors (e.g., policy sampling or ϵ-greedy selection), resulting in frequent collisions. From the agent's perspective, the scarcity of successful experiences significantly hinders training convergence. From the station's perspective, frequent collisions lead to queue congestion and substantially increase packet delay. To address these limitations, we propose an expanded action space that allows agents to wait for multiple time slots. We refer to this as the *multiple waiting actions mechanism*. Formally, the action space for each station is defined as $a \in \{0, 1, \ldots, N_w\}$, where

- $a = 0$ indicates transmitting a packet,
- $a > 0$ denotes waiting for a slots.

The proposed action mechanism is elaborated below:

1. Each station makes a decision (i.e., go to Step 2) only when its packet buffer is non-empty and the channel has been sensed idle for at least DIFS duration. Otherwise, it remains to sense the channel.
2. The station selects an action a by its equipped agent.
3. If $a = 0$, the station transmits a packet and then returns to Step 1.
4. If $a > 0$, the station initializes a counter $w = a$. This counter decrements by 1 during each idle slot. Once the channel becomes busy, the counter is frozen until the channel is sensed idle again for a DIFS period. When w decrements to 0, the station returns to Step 2.

Fig. 1 presents the above implementation process and Fig. 2 shows an example of two stations using this mechanism.

The appropriate value of N_w depends on both the number of stations N and the network traffic load $\hat{\lambda}$. As N or $\hat{\lambda}$ increases, collisions become more frequent, necessitating a larger N_w. To intelligently adapt to varying network traffic conditions, we allow N_w to be dynamically adjusted by the agent. Specifically, each agent selects N_w from a predefined set $\mathcal{S}_{N_w}$. We denote the maximum value of N_w as $N_w^{\max} = \max_{N_w \in \mathcal{S}_{N_w}} \{N_w\}$.

In the following, we analyze the relationships and distinctions between our proposed action mechanism and other approaches in existing MAC protocols.

- **Comparison with backoff mechanism**: The conventional backoff mechanism requires stations to select a backoff value BO from the contention window CW, i.e., $BO \in \{0, 1, \ldots, \text{CW} - 1\}$. Then, the station initiates a packet transmission after BO idle slots. In our mechanism, when the agent sequentially chooses to wait ($a = BO$) followed by transmission ($a = 0$), it exactly replicates the backoff behavior. However, unlike static backoff mechanisms, our approach allows the agent to adjust its action based on real-time observations of the "backoff" period, deriving a more flexible and intelligent access strategy. Moreover, when the wireless environment changes, the backoff mechanism requires careful tuning of CW to adapt to the new conditions. In contrast, our proposed mechanism can adapt automatically through interaction with the environment.
- **Comparison with binary-decision mechanism**: The proposed mechanism generalizes the binary-decision mechanism by expanding the waiting action space. When $N_w = 1$, our proposed mechanism reduces to the binary-decision case. When $N_w > 1$, our action mechanism enables more sophisticated channel access control: stations can select different waiting durations, which helps distribute transmission attempts over time and consequently reduces collision probability.

In general, the multiple waiting actions mechanism includes existing access mechanisms, such as the backoff mechanism and binary-decision mechanism. It provides a more general access paradigm, allowing a station to wait for multiple time slots before making each transmission decision.

B. Packet-based Agent Mechanism

In most RL-based MAC protocols, each station is regarded as an independent agent. To mitigate partial observability, the agent's state typically incorporates a history of its past channel access actions and observations over several time steps. However, this *station-based agent* formulation often includes outdated information, which may no longer accurately represent the current station state. As shown in Fig. 3, when a station makes a decision at current time t, the station-based agent's history contains past packet transmissions. Under unsaturated traffic conditions, stations frequently empty their packet buffers, leading to large intervals between packet arrivals. As a result, the history of the last packet transmissions is outdated.

To overcome this limitation, we propose a *packet-based agent* mechanism, where the head-of-line (HOL) packet is treated as an agent. In this mechanism, an episode begins when a packet becomes HOL and terminates when it is successfully transmitted or dropped. As shown in Fig. 3, the agent's history is recorded from the episode's start t_3 to the current time t. Below, we highlight the advantages of this approach over the station-based method:

- Unlike the station-based method, the packet-based agent excludes obsolete experiences from its history, maintaining only relevant action-observations.
- As is well established, reinforcement learning aims to maximize the expected cumulative reward. In the station-based method, cumulative rewards reflect long-term performance metrics of the station, which change slowly and thus provide delayed feedback. In contrast, the packet-based agent maximizes rewards within each episode, directly optimizing the packet delay.

In the following, we analyze the packet-based agent mechanism from the perspective of reinforcement learning. RL tasks are generally categorized as either episodic or continuing, depending on whether episodes terminate. The distributed channel access problem is modeled as a continuing task, as the objective is to optimize long-term network performance. From a theoretical perspective, the innovation of the proposed packet-based agent lies in reformulating a partially observable continuing task (2) into a series of episodic tasks (3), which effectively mitigates reward sparsity and accelerates policy convergence.

$$R_{\text{station}} = \lim_{T_s \to \infty} \sum_{t=0}^{T_s} \gamma^t r_t \tag{2}$$

$$R_{\text{packet}} = \sum_{t=0}^{T_p} \gamma^t r_t, \quad T_p \ll T_s. \tag{3}$$

Then, we analyze the relationship between the station-based objective in (2) and the packet-based objective in (3). By decomposing the continuing task into a sequence of packet-based episodes, the station-based return can be expressed as

$$R_{\text{station}} = \lim_{K \to \infty} \sum_{k=1}^{K} \gamma^{t_k} R_{\text{packet}}^{(k)}, \tag{4}$$

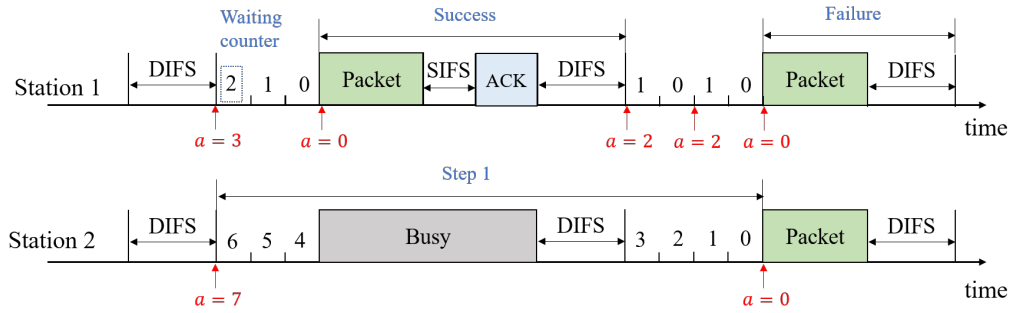


Fig. 2: An example of the distributed channel access scenario with two stations using the multiple waiting actions mechanism.

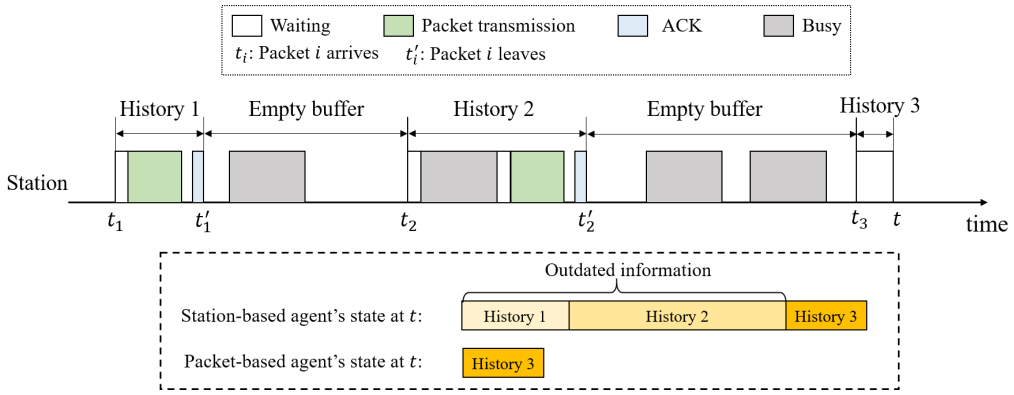


Fig. 3: The history comparison between the station-based and packet-based agent. Packets arrives at t_1, t_2, and t_3. Two of them are successfully transmitted at t_1' and t_2'. The current time is denoted as t.

where t_k denotes the starting time of the k-th packet and $R^{(k)}_{\text{packet}}$ is its episodic return. This expression indicates that the station-based discounted return equals a weighted sum of packet-based returns, with weights γ^{t_k}. When the discount factor satisfies $\gamma \approx 1$, the variation of γ^{t_k} across packets becomes negligible. Under this condition, maximizing the expected packet-based return approximates maximizing the station-based long-term objective, thereby providing theoretical justification for the packet-based agent formulation.

V. RL Formulation and Algorithm

In this section, we propose the SAC-MA algorithm that leverages the mechanisms introduced in the previous section. The remainder of the section is organized as follows: we first present the RL formulation and the neural network architectures of the actor and critic. Then, we elaborate on the implementation details of the SAC-MA protocol.

A. RL Formulation

The system model can be modeled as a multi-agent task, where the HOL packet in each station $i \in \mathcal{N} \equiv \{1, 2, \ldots, N\}$ is regarded as an agent. At each time step, each agent i chooses an action $a^i \in \mathcal{A}$. After that, it receives a *reward* r^i and a *local observation* $z^i \in \mathcal{Z}$. To mitigate the partially observable issue, agents estimate the state by adopting their individual *action-observation history* $\tau^i \in \mathcal{T} \equiv (\mathcal{Z} \times \mathcal{A})^{L_h}$, which contains the action-observation pairs over past L_h time steps. Agent i learns a policy $\pi_{\theta^i}(a^i|\tau^i)$ with the parameters θ^i to maximize the objective $J(\theta^i) = \mathbb{E}_{\pi_{\theta^i}}\left[\sum_{t=0}^{T} \gamma^t r_t^i\right]$, where T is the time horizon and $\gamma \in (0, 1]$ is a discount factor. In this paper, a step is defined as the interval between two consecutive decisions. Since each agent is considered to train in a distributed manner, we omit the superscript i of the variables hereafter. We then provide detailed descriptions of each agent's local observation, action-observation history, and reward.

1) Local Observation: The local observation $z = [o, \bar{D}, \bar{D}_o]$ consists of three parts:

- o: We characterize the channel observation using a tripe o that represents the proportion of the numbers of successful transmissions, collisions, and idle slots at each step. Formally, o is defined as:

$$o = \left[\frac{N_s}{N_{\text{sum}}}, \frac{N_f}{N_{\text{sum}}}, \frac{N_i}{N_{\text{sum}}}\right], \tag{5}$$

where N_s, N_f, and N_i represent the number of successful transmissions, failed transmissions, and idle slots, respectively, and $N_{\text{sum}} = N_s + N_f + N_i$. In particular, when the station attempts to transmit the HOL packet, its channel observation is either $[1, 0, 0]$ (success) or $[0, 1, 0]$ (failure). In practice, each station updates N_s and N_f based on ACK reception. When the channel's busy state ends, $N_s \leftarrow N_s + 1$ if an ACK is received; otherwise, $N_f \leftarrow N_f + 1$. Take the station 2 in Fig. 2 for example. At step 1, the values of N_s, N_f, and N_i are $1, 0, 7$. Consequently, the channel observation is $[0.125, 0, 0.875]$.

- $\bar{D}$: We define D as the sojourn time of the HOL packet, i.e., the duration from its generation to the current time. The normalized waiting time is computed as $\bar{D} = \frac{D}{D_{\max}}$, where $D_{\max}$ represents the maximum end-to-end delay among the most recent 10^3 successfully transmitted packets by the station itself.
- $\bar{D}_o$: Since each station cannot acquire the packet buffer information of others, we adopt the *delay to last transmission* (D2LT) [3] to represent the state of other stations. The D2LT of other stations D_o is defined as the duration since the last successful transmission by any other station. Its normalized form is given by $\bar{D}_o = \frac{D_o}{D_{\max}}$.

2) Action-observation History: The action-observation c can be denoted as

$$c \triangleq [\tilde{a}, z] = [\tilde{a}, o, \bar{D}, \bar{D}_o], \tag{6}$$

where $\tilde{a}$ is normalized by the $N_w^{\max}$ to the range $[0, 1]$, i.e., $\tilde{a} = \frac{a}{N_w^{\max}}$. The action-observation at step j is denoted as c_j by introducing the subscript j. Leveraging the packet-based agent mentioned in Section IV-B, the action-observation history at current step t_s can be expressed as

$$\tau_{t_s} \triangleq [c_0, c_1, \ldots, c^i_{t_s-1}], \tag{7}$$

which includes the history of action-observation from the time the packet becomes HOL ($j = 0$) to the current time step ($j = t_s$). If the length of the history exceeds L_h, it is truncated to the most recent L_h action-observation pairs.

3) Reward Function for Action a: In this work, we focus on optimizing the end-to-end (E2E) delay, which is composed of access delay and waiting time:

- **Access delay** is defined as the time from when a packet becomes head-of-line to its successful transmission.
- **Waiting time** is the time from a packet's arrival to when it becomes the head-of-line packet.

To optimize access delay and waiting time, we design separate reward functions below.

The reward for access delay is defined as:

$$r_a = \begin{cases} 1, & \text{if success,} \\ -\dfrac{T_{\text{col}}}{T_{\text{succ}}}, & \text{if failure,} \\ -\dfrac{aT_s}{T_{\text{succ}}}, & \text{if waiting for } a \text{ slots,} \end{cases} \tag{8}$$

where T_{col} and T_{succ} are the duration of a failed transmission and successful transmission, respectively. Below, we introduce the reward design in detail. If the HOL packet is successfully transmitted, a positive reward of $+1$ is assigned. In the case of a transmission failure, the access delay of the HOL packet increases by the duration of collision T_{col}. To reflect this cost, we normalize T_{col} by T_{succ} and assign a negative reward $-\frac{T_{\text{col}}}{T_{\text{succ}}}$ to penalize the agent. When the agent chooses to wait, it wastes a slots, resulting in a negative reward of $-\frac{aT_s}{T_{\text{succ}}}$.

The reward for waiting time is defined as follows:

$$r_q = \begin{cases} -\dfrac{T_{\text{col}}(L_b - 1)}{T_{\text{succ}} L}, & \text{if failure,} \\ -\dfrac{aT_s(L_b - 1)}{T_{\text{succ}} L}, & \text{if waiting for } a \text{ slots,} \\ 0, & \text{otherwise,} \end{cases} \tag{9}$$

where L_b represents the number of packets in the buffer, and L is the maximum buffer size. When the HOL packet is unsuccessfully transmitted, the waiting time for all packets except the HOL packet increases by $T_{\text{col}}(L_b - 1)$. Similarly, waiting for a slots increases the waiting time by at least $aT_s(L_b - 1)$. We normalize the increasing waiting time by $T_{\text{succ}} L$ and assign negative rewards to penalize the agent.

To reduce the 95th percentile of the E2E delay distribution, we define an additional reward as:

$$r_{95} = -\min\{1, \frac{D}{D_{95}}\}, \tag{10}$$

where D_{95} represents the 95th percentile of the E2E delay distribution. The value of D_{95} is calculated based on the most recent 10^3 successfully transmitted packets by the station itself. When the sojourn time D of the HOL packet approaches 95th-percentile delay D_{95}, the agent receives a larger penalty. If D exceeds D_{95}, the reward r_{95} is set to -1.

The final reward for action a is a weighted sum of the above components:

$$r = \omega(r_a + r_q) + (1 - \omega) r_{95}, \tag{11}$$

where $\omega \in [0, 1]$ is a weight that balances the contributions of the different reward components. As previously introduced, accumulating the above reward at each step reflects the delay performance with a negative sign. By maximizing the cumulative reward, the agent learns a policy that reduces delay.

4) Reward Function for N_w: Since N_w is selected by the agent, we design a reward function to guide its learning:

$$\tilde{r} = \begin{cases} +1, & \text{if success,} \\ -1, & \text{if failure,} \\ 0, & \text{otherwise.} \end{cases} \tag{12}$$

This reward encourages the agent to select the value of N_w that leads to more successful transmissions while penalizing those that result in failures.

B. Actor-Critic Architecture

In this work, we adopt an actor-critic framework [36] to learn an access strategy. Each agent is equipped with its own actor and critic networks. The actor network of each agent generates a policy π for action selection, while the critic network guides the learning process of the corresponding actor. The architectures of these networks are illustrated in Fig. 5.

1) Actor: The architecture of the actor network is illustrated in Fig. 5a. For each agent, the action-observations in τ are sequentially fed into a gated recurrent unit (GRU) network to extract the temporal feature. The hidden state of the final GRU layer is then passed through a fully connected (FC) layer with a leaky rectified linear unit (Leaky ReLU) activation function. The output of this FC layer is then forwarded to two distinct FC layers, "FC $|\mathcal{A}|$" layer and "FC $|\mathcal{S}_{N_w}|$" layer, respectively. The FC $|\mathcal{S}_{N_w}|$ layer with softmax function outputs the distribution of $\tilde{\pi}(N_w|\tau)$, which is sampled for selecting N_w. The FC $|\mathcal{A}|$ layer outputs a vector y of length $|\mathcal{A}| = N_w^{\max} + 1$, ensuring that the actor network can accommodate the maximum possible value of N_w. Note that

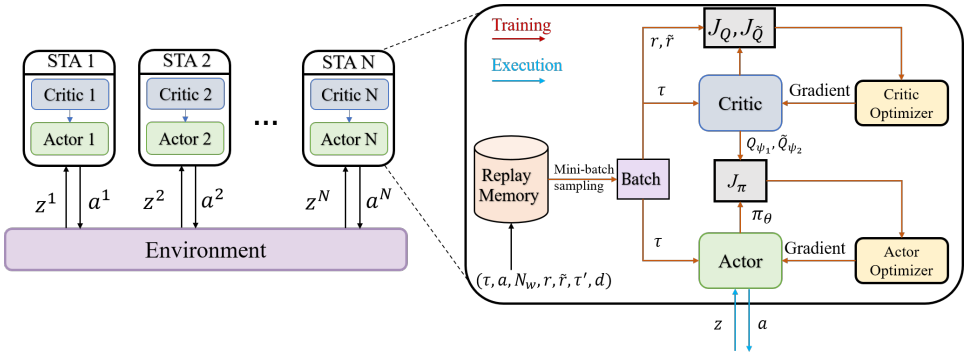
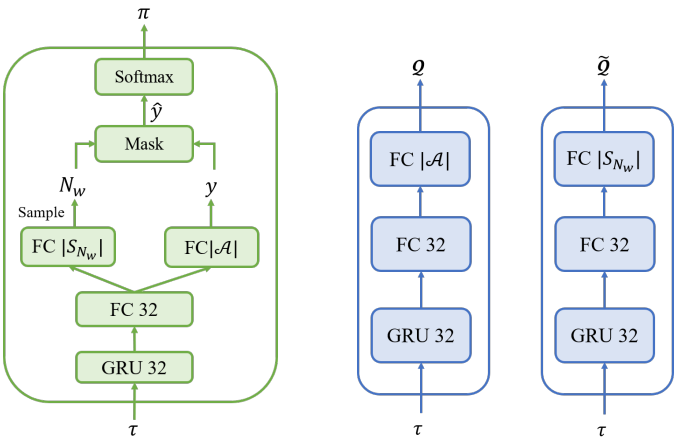


Fig. 4: The training and execution processes of the SAC-MA protocol.



(a) Actor network (b) Critic network

Fig. 5: Network architecture of the actor-critic framework. The number following "FC" or "GRU" denotes the number of neurons in the corresponding layer.

Algorithm 1 Multiple Waiting Actions Mechanism

procedure MULTIPLE WAITING ACTIONS(w, θ, τ)
  if Packet buffer is not empty **and** Channel is sensed idle for more than DIFS duration **then**
    if $w = 0$ **then**
      Select $N_w \sim \tilde{\pi}_\theta(\cdot|\tau)$
      Select an action $a \sim \pi_\theta(\cdot|\tau)$
      if $a > 0$ **then**
        $w \leftarrow a$
    else
      $w \leftarrow w - 1$
  if $a = 0$ **then**
    Transmit a packet
  else
    Perform carrier sensing for a time slot
  return a, N_w

the action a belongs to the set $\{0, 1, \ldots, N_w\}$. To prevent the selection of illegal actions (i.e., a exceeding the chosen N_w), we apply a masking operation:

$$\hat{y}_k = \begin{cases} -10^9, & \text{if } k \geq (N_w + 1), \\ y_k, & \text{otherwise}, \end{cases} \tag{13}$$

where y_k and $\hat{y}_k$ are the k-th element in the vector y and $\hat{y}$, respectively. After masking, a softmax function is applied to generate the policy distribution $\pi(a|\tau)$, from which an action a is sampled:

$$a \sim \pi(a|\tau) = \text{softmax}\left(\hat{y}\right). \tag{14}$$

Since $\hat{y} = -10^9$ is used for illegal actions, their corresponding softmax probabilities are approximately zero.

2) Critic: As shown in Fig. 5b, we employ two distinct critic networks to guide the selection of a and N_w, depicted on the left and right sides of the figure, respectively. For the critic associated with a, the network first encodes the trajectory τ using GRU. The GRU output is subsequently passed through two FC layers to compute the action-value functions $\boldsymbol{\mathcal{Q}} \triangleq \{Q(\tau, a)|a \in \mathcal{A}\}$, where $Q(\tau, a)$ represents the estimated value of taking action a given the history τ. The first FC layer applies the leaky ReLU activation function, while the second FC layer uses a linear activation. The architecture of the critic network for N_w is similar to that for a, with the difference being that the output layer contains $|\mathcal{S}_{N_w}|$ neurons. This critic network outputs the action-value functions $\tilde{\boldsymbol{\mathcal{Q}}} \triangleq \{\tilde{Q}(\tau, N_w)|N_w \in \mathcal{S}_{N_w}\}$ for different N_w.

C. SAC-MA Algorithm

In this section, we introduce the process of our proposed SAC-MA protocol. We adopt a fully decentralized multi-agent reinforcement learning framework, where both the training and execution processes are decentralized. These processes are presented in Fig. 4. The pseudo-code of multiple waiting actions mechanism and the SAC-MA algorithm are summarized in Algorithm 1 and Algorithm 2, respectively. Employing the experience replay mechanism [37], each agent is equipped with a replay memory to store the experiences from past steps. At each step, an experience tuple $(\tau, a, N_w, r, \tilde{r}, \tau', d)$ is stored in the replay memory, where τ' is the transited history after executing a. The variable d denotes whether the episode has terminated: $d = 1$ if the episode ends, and $d = 0$ otherwise.

In the following, we detail the training process. First, each agent samples a batch of B experiences as E to update the actor-critic networks. In each station, the network parameters of the actor are denoted by θ. The network parameters of

Algorithm 2 SAC-MA Algorithm

Initialize parameters: $t \leftarrow 0$, $\tau \leftarrow \tau_0$, $z_0 \leftarrow \mathbf{0}$, $\bar{\psi_1} \leftarrow \psi_1, \bar{\psi_2} \leftarrow \psi_2, w \leftarrow 0$.
Initialize the replay memory $\mathcal{D}$.
while time $t < T_{\text{sim}}$ **do**
  // Execution process
  $a, N_w =$ MULTIPLE WAITING ACTIONS(w, θ, τ)
  Channel provides feedback to the station
  // Training process
  if The station selects an action using $\pi_\theta(\cdot|\tau)$ **then**
    Get local observation z
    Compute τ' from τ, a, z
    Compute r and $\tilde{r}$ from Eq. (11) and Eq. (12)
    $d \leftarrow$ whether the episode has terminated
    Store $(\tau, a, N_w, r, \tilde{r}, \tau', d)$ to $\mathcal{D}$
    Randomly sample B experiences from $\mathcal{D}$ as E
    // Update network parameters
    $\psi_1 \leftarrow \psi_1 - \beta_\psi \hat{\nabla}_{\psi_1} J_Q(\psi_1)$
    $\psi_2 \leftarrow \psi_2 - \beta_\psi \hat{\nabla}_{\psi_2} J_{\tilde{Q}}(\psi_2)$
    $\theta \leftarrow \theta - \beta_\theta \hat{\nabla}_\theta J_\pi(\theta)$
    $\alpha_1 \leftarrow \alpha_1 - \beta_\alpha \hat{\nabla}_{\alpha_1} J_{\alpha_1}(\alpha_1)$
    $\alpha_2 \leftarrow \alpha_2 - \beta_\alpha \hat{\nabla}_{\alpha_2} J_{\alpha_2}(\alpha_2)$
    $\bar{\psi_1} \leftarrow \eta\psi_1 + (1-\eta)\bar{\psi_1}$
    $\bar{\psi_2} \leftarrow \eta\psi_2 + (1-\eta)\bar{\psi_2}$
    // Packet-based agent mechanism
    if $d = 1$ **then**
      $\tau \leftarrow \tau_0$
    else
      $\tau \leftarrow \tau'$
  Update time t

the critic for a and N_w are denoted by ψ_1 and ψ_2. The critic networks are equipped with their corresponding target network [37], whose parameters are represented by $\bar{\psi_1}$ and $\bar{\psi_2}$. We employ the discrete-action version of the soft actor-critic method [38] to update the network parameters. SAC algorithm aims to simultaneously maximize both expected cumulative reward (i.e., return) and policy entropy:

$$J(\theta) = \mathbb{E}_{\pi_\theta}\left[\sum_{t=0}^{T}\left(\gamma^t r_t + \alpha\mathcal{H}(\cdot|\tau)\right)\right], \tag{15}$$

where $\mathcal{H}(\cdot|\tau)$ is the entropy of the policy and α is the temperature parameter that balances entropy maximization and returns. Distinct temperature parameters, α_1 and α_2, are employed for training the action a and the variable N_w, respectively. This encourages exploration by promoting more diverse action choices. The loss function of the critic for a is defined in the form of mean square error:

$$J_Q(\psi_1) = \frac{1}{B}\sum_{e\in E}\left[Q_{\psi_1}(\tau, a) - \left(r + \gamma(1-d)V_{\bar{\psi_1}}(\tau')\right)\right]^2, \tag{16}$$

where

$$V_{\bar{\psi_1}}(\tau') = \sum_{\tilde{a}\in\mathcal{A}}\pi_\theta(\tilde{a}|\tau')\left[Q_{\bar{\psi_1}}(\tau', \tilde{a}) - \alpha_1\log(\pi_\theta(\tilde{a}|\tau'))\right] \tag{17}$$

TABLE I: System parameters

Parameters	Value
PHY header	$36\mu s$
MAC header	26 bytes
ACK	(14 bytes)/R_B + PHY header
Time slot length	$9\mu s$
SIFS	$16\mu s$
DIFS	$34\mu s$
Packet payload length	2304 bytes
Basic rate R_B	6 Mbps
Transmission rate R_T	16 Mbps
Packet buffer size L	50
Capture threshold μ	0.1
Mean received SNR ρ	20 dB
Maximum retransmission limit	10

is the soft state value function. Similarly, the loss function of the critic for N_w is defined as:

$$J_{\tilde{Q}}(\psi_2) = \frac{1}{B}\sum_{e\in E}\left[\tilde{Q}_{\psi_2}(\tau, N_w) - \left(\tilde{r} + \gamma(1-d)\tilde{V}_{\bar{\psi_2}}(\tau')\right)\right]^2, \tag{18}$$

where

$$\tilde{V}_{\bar{\psi_2}}(\tau') = \sum_{\tilde{N_w}\in\mathcal{S}_{N_w}}\tilde{\pi}_\theta(\tilde{N_w}|\tau')\left[\tilde{Q}_{\bar{\psi_2}}(\tau', \tilde{N_w}) - \alpha_2\log(\tilde{\pi}_\theta(\tilde{N_w}|\tau'))\right]. \tag{19}$$

The parameters of the actor can be learned by minimizing

$$J_\pi(\theta) = \frac{1}{B}\sum_{e\in E}[\sum_{\tilde{a}\in\mathcal{A}}\pi_\theta(\tilde{a}|\tau)[\alpha_1\log(\pi_\theta(\tilde{a}|\tau)) - Q_{\psi_1}(\tau, \tilde{a})] + \sum_{\tilde{N_w}\in\mathcal{S}_{N_w}}\tilde{\pi}_\theta(\tilde{N_w}|\tau)[\alpha_2\log(\tilde{\pi}_\theta(\tilde{N_w}|\tau)) - \tilde{Q}_{\psi_2}(\tau, \tilde{N_w})]]. \tag{20}$$

The automating entropy adjustment mechanism is used to adjust learnable temperature parameters α_1 and α_2 by minimizing

$$J_{\alpha_1}(\alpha_1) = \frac{1}{B}\sum_{e\in E}\sum_{\tilde{a}\in\mathcal{A}}\pi_\theta(\tilde{a}|\tau)[-\alpha_1(\log(\pi_\theta(\tilde{a}|\tau)) + \bar{H}_1)], \tag{21}$$

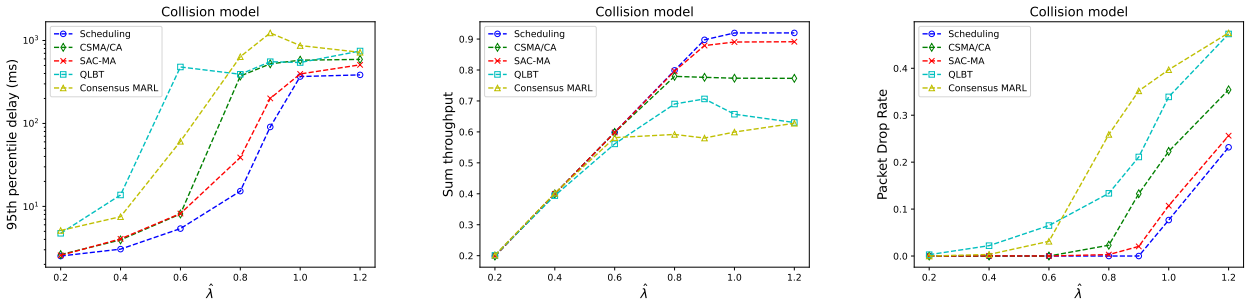
and

$$J_{\alpha_2}(\alpha_2) = \frac{1}{B}\sum_{e\in E}\sum_{\tilde{N_w}\in\mathcal{S}_{N_w}}\tilde{\pi}_\theta(\tilde{N_w}|\tau)[-\alpha_2(\log(\tilde{\pi}_\theta(\tilde{N_w}|\tau)) + \bar{H}_2)], \tag{22}$$

respectively. $\bar{H}_1$ and $\bar{H}_2$ are predefined target values for policy entropy. The soft update method is applied to update the parameters of target networks, i.e., $\bar{\psi_1} \leftarrow \eta\psi_1 + (1-\eta)\bar{\psi_1}$ and $\bar{\psi_2} \leftarrow \eta\psi_2 + (1-\eta)\bar{\psi_2}$, where η represents the soft-update factor.

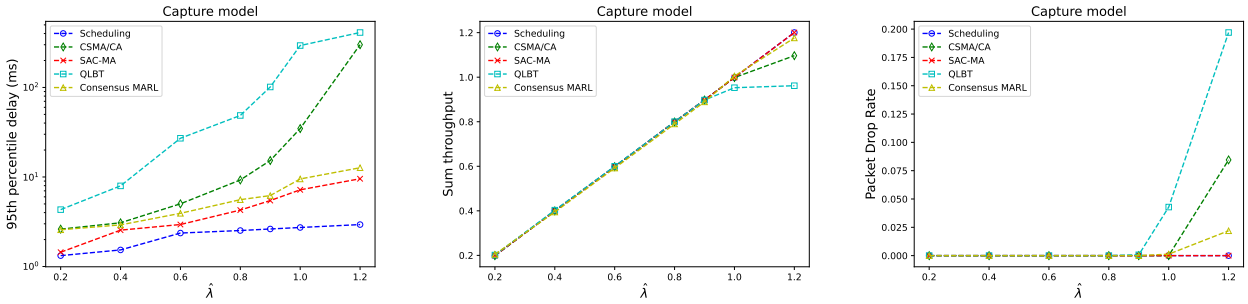
VI. Simulation Results

In this section, we first introduce the simulation setup and performance metrics. We then provide detailed performance evaluations under static and dynamic scenarios. After that, we evaluate the performance in heterogeneous network settings and perform the ablation experiments.



(a) 95th percentile delay (b) Sum throughput (c) Packet drop rate

Fig. 6: The performance comparison under static scenarios for the collision model with different network traffic loads. The number of stations is fixed at 5.



(a) 95th percentile delay (b) Sum throughput (c) Packet drop rate

Fig. 7: The performance comparison under static scenarios for the capture model with different network traffic loads. The number of stations is fixed at 5.

A. Simulation Setup

We adopt the system parameters specified in the IEEE 802.11be standard [39], which is presented in Table I. Each station transmits a packet composed of the physical (PHY) header, MAC header, and payload. The duration of packet transmission, T_p, is calculated as:

$$T_p = \text{PHY header} + \frac{\text{MAC header} + \text{Payload}}{R_T}, \tag{23}$$

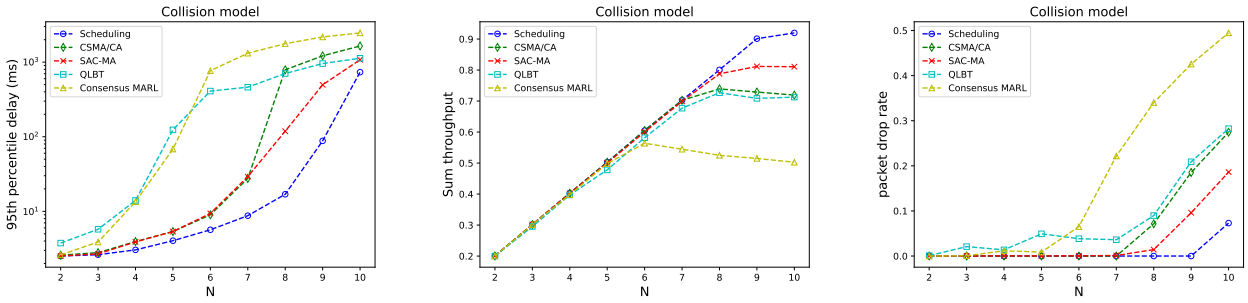
where the MAC header and payload are transmitted with the transmission rate R_T. According to the standard, the PHY header lasts for $36\mu s$ and the MAC Header consists of 26 bytes. The payload is set to 2304 bytes, which is the maximum size for a MAC service data unit. The ACK frame is transmitted with the basic rate R_B. The packet arrival of each station follows a Bernoulli distribution with the same arrival rate λ. At each interval of T_p, each station generates a new packet with probability λ. The aggregate input rate is $\hat{\lambda} = N\lambda$. Each station is equipped with a packet buffer with a maximum capacity of 50 packets. The maximum retransmission limit is set to 10. The capture threshold and the mean received SNR are set to 0.1 and 20dB, respectively. The simulation time is set to $T_{\text{sim}} = 50$ seconds by default.

To provide a comprehensive comparison, we compare our proposed SAC-MA algorithm with baselines including scheduling, CSMA/CA, and QLBT method. The parameter settings of these methods are described in detail below.

TABLE II: Hyper-parameters of SAC-MA

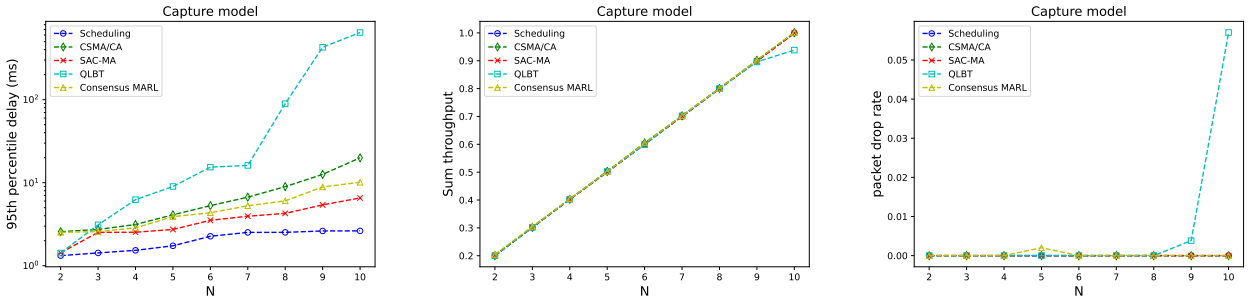
Parameters	Value
The set of N_w	$\{1, 4, 8\}$
Maximum length of the history L_h	40
The capacity of the replay memory $\mathcal{D}$	1000
Learning rate β_θ, β_ψ	5×10^{-4}
Learning rate β_α	1×10^{-2}
The initial value of α	0.5
Target entropy H_1	$-0.4\ln(1/(N_w + 1))$
Target entropy H_2	$-0.4\ln(1/(\lvert\mathcal{S}_{N_w}\rvert))$
Batch size B	16
Discount factor γ	0.99
Soft update factor η	0.01
Weight in the reward function ω	0.8

1) SAC-MA: We first introduce the parameters of neural networks. As shown in Fig. 5a, both the GRU and the first FC layer in actor and critic networks contain 32 neurons. The numbers of their final layers are set to $|\mathcal{A}| = 9$ or $|\mathcal{S}_{N_w}| = 3$, where S_{N_w} is set to $\{1, 4, 8\}$. The maximum length of action-observation history is set to 40. The replay memory $\mathcal{D}$ stores the recent 1000 experiences. Then, the training parameters in the SAC method are given as follows. The parameters in actor-critic networks and learnable temperature parameter α are updated using the respective Adam optimizers with learning rates $\beta_\theta = \beta_\psi = 5 \times 10^{-4}$ and $\beta_\alpha = 1 \times 10^{-2}$. α is initially set to 0.5, and $\bar{H}_1$ and $\bar{H}_2$ are set to $-0.4\ln(1/|\mathcal{A}|)$ and



(a) 95th percentile delay (b) Sum throughput (c) Packet drop rate

Fig. 8: The performance comparison under static scenarios for the collision model with different numbers of stations. The packet arrival rate of each station is fixed at $\lambda = 0.1$.



(a) 95th percentile delay (b) Sum throughput (c) Packet drop rate

Fig. 9: The performance comparison under static scenarios for the capture model with different numbers of stations. The packet arrival rate of each station is fixed at $\lambda = 0.1$.

$-0.4\ln(1/|\mathcal{S}_{N_w}|)$, respectively. The batch size B, discount factor γ, and soft update factor η are set to 16, 0.99, 0.01, respectively. The weight ω in the reward function is set to 0.8. The hyper-parameters used in SAC-MA are summarized in Table II.

2) Scheduling: We assume an ideal access strategy for the scheduling method, with full knowledge of all stations' packet buffer states and channel fading coefficients, and no scheduling overhead. In the collision model, the scheduler randomly selects a station with a non-empty packet buffer to transmit. In the capture model, the scheduler evaluates all possible access actions and selects the one that maximizes the number of successful transmissions.

3) CSMA/CA: We consider the basic access mode of CSMA/CA. According to the standard, the initial backoff window size W and the maximum backoff phase K are set to 16 and 6, respectively.

4) QLBT: As a RL-based MAC protocol, the QLBT algorithm is introduced as a baseline. Its hyper-parameters setting is consistent with those in the literature [3].

5) Consensus MARL: We adopt the algorithm from [25], referred to as *Consensus MARL*, as an additional baseline. The hyperparameter settings are consistent with those reported in [25], except that the normalization factor w_0 is set to $= 1/600$ instead of $1/60$ since the packet length in our work is larger than that in Consensus MARL.

B. Performance Metrics

Below, we present the performance metrics used to evaluate our algorithm and other baselines.

1) 95th Percentile Delay: The E2E delay is defined as the time interval between the generation of a packet and the reception of its corresponding ACK. The 95th percentile delay is computed as the value below which 95% of the E2E delays of successfully delivered packets fall.

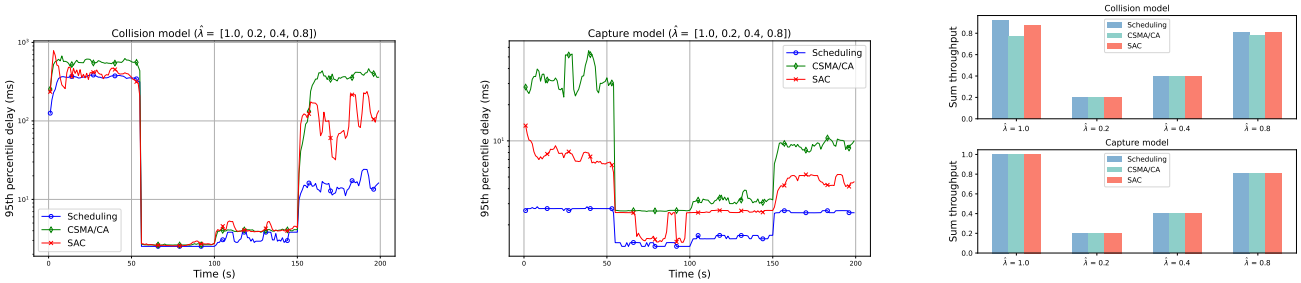
2) Sum Throughput: The sum throughput is defined as the ratio of the total transmission time of all successfully transmitted packets to the simulation time T_{sim}, which is calculated as:

$$Th_{\text{sum}} = \frac{N_{\text{succ}} \times T_p}{T_{\text{sim}}}, \tag{24}$$

where N_{succ} represents the number of packets successfully transmitted.

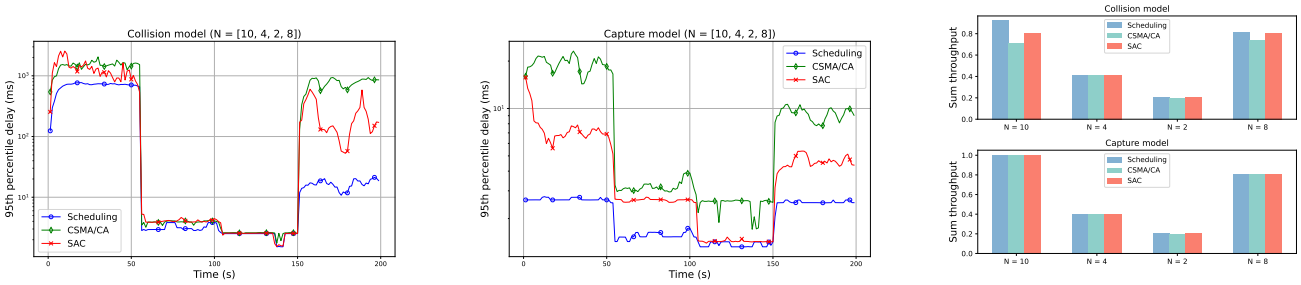
3) Packet Drop Rate: A packet is dropped under either of the following conditions: (a) it arrives when the packet buffer is full, or (b) its number of transmission failures exceeds the maximum retransmission limit. Let N_{drop} denote the number of dropped packets over the entire simulation. The packet drop rate (PDR) is calculated as:

$$\text{PDR} = \frac{N_{\text{drop}}}{N_{\text{drop}} + N_{\text{succ}}}. \tag{25}$$



(a) Delay performance (collision model) (b) Delay performance (capture model) (c) Throughput performance

Fig. 10: The performance comparison under dynamic traffic scenarios. The number of stations is fixed at 5.



(a) Delay performance (collision model) (b) Delay performance (capture model) (c) Throughput performance

Fig. 11: The performance comparison under dynamic network sizes. The packet arrival rate of each station is fixed at $\lambda = 0.1$.

C. Static Scenario

In the static scenario, system parameters such as the network size N and network traffic load $\hat{\lambda}$ are fixed throughout each simulation. We evaluate the SAC-MA algorithm and other baseline methods under static scenarios with different system parameters for the same simulation time, i.e., T_{sim} seconds.

First, we present simulation results under different traffic loads with a fixed network size of $N = 5$, as shown in Fig. 6 and Fig. 7 for the collision and capture models, respectively. A logarithmic scale is used for the Y axis in the delay performance to better visualize the wide range of values. For the 95th percentile delay, SAC-MA outperforms the QLBT and Consensus MARL algorithm. Compared with CSMA/CA, SAC-MA reduces delay by 27.9% and 56.5% in the collision and capture models, respectively. Furthermore, compared to CSMA/CA, SAC-MA reduces the delay performance gap with the scheduling method by approximately 47.6% and 76.5% under the collision and capture models, respectively. In terms of throughput, all methods achieve the target sum throughput $Th_{\text{sum}} = \hat{\lambda}$ under unsaturated conditions. As the traffic load increases, the network becomes saturated and the sum throughput does not increase anymore. The simulation results indicate that SAC-MA achieves higher saturation throughput than all baselines in the collision model. Moreover, SAC-MA exhibits a lower packet drop rate under saturated conditions.

Then, we consider static scenarios with different numbers of stations N while fixing the packet arrival rate per station at $\lambda = 0.1$. The number of stations is varied from 2 to 10. The simulation results for the collision and capture model are shown in Fig. 8 and Fig. 9, respectively. Compared to the CSMA/CA, QLBT and Consensus MARL method, the proposed SAC-MA algorithm achieves lower 95th percentile delay and packet drop rate, as well as higher sum throughput, when the number of stations is large.

D. Dynamic Scenario

To evaluate the adaptation ability of SAC-MA, we consider dynamic wireless networks where system parameters change in real time. In these scenarios, agents using the SAC-MA algorithm are continuously trained throughout the entire simulation. Given that the QLBT and Consensus MARL method perform poorly in static scenarios, our evaluation focuses on SAC-MA, CSMA/CA, and the scheduling method. We first examine performance under dynamic traffic load conditions. The scenario is configured as follows: the number of stations is set to 5, and the simulation is divided into four intervals, each of which lasts 50 seconds. The network traffic load $\hat{\lambda}$ in each interval is set to 1, 0.2, 0.4, and 0.8, respectively. The simulation results under the collision model and capture model are presented in Fig. 10a and Fig. 10b, respectively, where each data point represents the 95th percentile delay computed from successfully transmitted packets over the past 5 seconds. The results demonstrate that SAC-MA adapts effectively to changing traffic conditions and outperforms CSMA/CA during most of the simulation time. Fig. 10c shows the sum throughput for each interval. Under unsaturated traffic conditions, the sum throughput of all methods matches the offered load, i.e., $Th_{\text{sum}} = \hat{\lambda}$. However, in saturated conditions, such as

when $\hat{\lambda} = 1.0$ in the collision model, the SAC-MA algorithm achieves higher throughput than CSMA/CA. Next, we evaluate performance under dynamic network size with fixed $\lambda = 0.1$. The simulation is also divided into four intervals, each of which lasts 50 seconds. The number of stations in each interval is 10, 4, 2, and 8, respectively. As illustrated in Fig. 11, SAC-MA successfully adapts to changes in network size with respect to delay performance. Moreover, under the collision model, it achieves higher sum throughput than CSMA/CA, particularly when the number of stations is large (e.g., $N = 8$ or $N = 10$).

E. Heterogeneous Network

In this subsection, we consider heterogeneous networks where SAC-MA stations coexist with stations using the CSMA/CA protocol. To evaluate fairness between these two protocols, we adopt the 3GPP fairness metric, which requires that the introduction of a new protocol should not degrade the performance (e.g., throughput and delay) of an existing Wi-Fi network more than the addition of another Wi-Fi network on the same channel. We assume that all legacy Wi-Fi stations use the CSMA/CA protocol with the same parameters specified in Section VI-A3. Since the CSMA/CA protocol employs a binary exponential backoff mechanism, the contention window increases upon collisions, leading to a longer average backoff time for CSMA/CA stations compared to SAC-MA stations. To protect these incumbent CSMA/CA stations, we impose a regulation on SAC-MA stations: each of them must wait for $N_h = 8$ slots following the DIFS duration whenever a packet becomes head-of-line. This operating mode is referred to as heterogeneous mode. In practice, an access point can broadcast a beacon frame to instruct SAC-MA stations to switch to this mode. In our simulations, we evaluate the throughput and delay in networks consisting of N SAC-MA stations and M CSMA/CA stations. To assess 3GPP fairness, we compare this setup with an alternative configuration in which the SAC-MA stations are replaced by an equivalent number of CSMA/CA stations, denoted as CSMA/CA-2. Simulations are conducted with various combinations of N and M under the collision model and capture model, with results presented in Fig. 12a, 12b, 13a, and 13b. The figures show the 95th percentile delay and throughput performance for both the CSMA/CA subset and the entire network. The results indicate that CSMA/CA stations coexisting with SAC-MA stations achieve throughput and delay performance that are no worse than when coexisting with CSMA/CA-2 stations, thereby satisfying the 3GPP fairness criterion. Furthermore, the inclusion of SAC-MA stations slightly reduces the overall network delay.

To analyze the impact of N_h on delay performance, we conduct simulations for a scenario with $N = 5$ and $M = 10$, varying the value of N_h. The simulation results are shown in Fig. 12c and Fig. 13c for the collision model and capture model, respectively. The results reveal that increasing N_h reduces the 95th percentile delay of CSMA/CA stations, as SAC-MA stations defer for more time slots before transmission, thereby providing CSMA/CA stations with more transmission opportunities. The 3GPP fairness for delay is satisfied when the delay of CSMA/CA stations falls below the dashed line, which represents their delay when coexisting with CSMA/CA-2 stations. The results indicate that setting $N_h = 8$ is an appropriate choice, ensuring 3GPP fairness while reducing overall network delay.

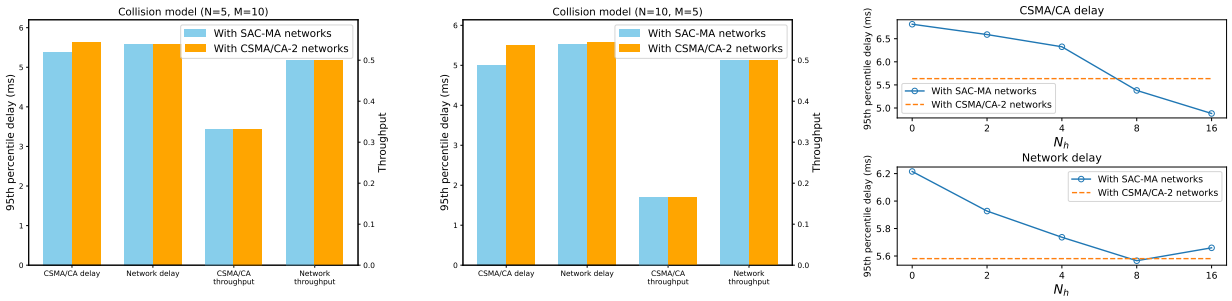
F. Discussions on Proposed Mechanisms

To discuss the impact of the proposed mechanisms in SAC-MA, we conduct ablation experiments by removing each mechanism individually. We first focus on the multiple waiting actions mechanism. A variant of SAC-MA referred to as "SAC-MA ($N_w = 1$)", is evaluated, where the number of waiting actions is fixed to 1. This variant corresponds to the algorithm using the binary-decision mechanism. Simulations are conducted under the collision model with a fixed arrival rate $\lambda = 0.1$ and varying network sizes: 2, 4, 6, and 8. The delay performance comparison between SAC-MA and its variant is illustrated in the upper plot of Fig. 14. The results show that restricting N_w to 1 leads to a significant increase in delay. This degradation is primarily attributed to the reduced waiting action space, which results in a higher collision probability, as depicted in the lower plot of Fig. 14.

Next, we evaluate the effectiveness of the packet-based agent mechanism. In contrast to the packet-based agent, the station-based agent maintains a history of multiple past packets rather than only the head-of-line packet. Additionally, episodes for the station-based agent are not terminated until the end of the entire simulation, meaning that the termination indicator d remains 0. The simulation setup is identical to that used in the previous mechanism analysis. As shown in Fig. 15, the packet-based method achieves better performance in terms of the 95th percentile delay. To further compare the history freshness between the two approaches, we introduce a metric called *age of history*, defined as the sum of the ages of all action-observation pairs, where the age of each pair is the time elapsed since its generation. We compute the mean age of history used in decision-making over the entire simulation. The lower subplot of Fig. 15 shows that the packet-based agent maintains a significantly lower mean age of history compared to the station-based agent, indicating that it relies on fresher information.

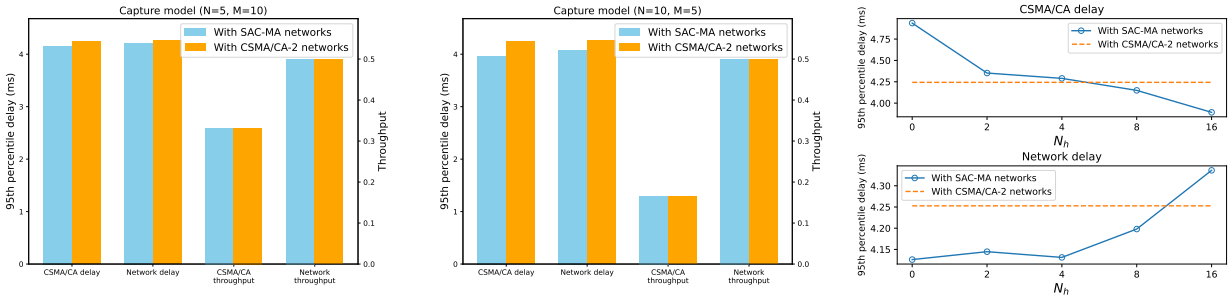
VII. Conclusion and Future Work

In this work, we propose a RL-based MAC protocol, named SAC-MA, to optimize the delay performance in the distributed channel access scenario. To reduce the collision among stations, we introduce the multiple waiting actions mechanism, which expands the action space by allowing agents to wait for multiple time slots. Moreover, we design a packet-based agent mechanism that treats the head-of-line packet as an independent agent, enabling the use of fresher historical information. We conduct extensive simulations to evaluate the performance of the proposed algorithm against baseline methods across various network scenarios. The results demonstrate that SAC-MA outperforms both the QLBT algorithm and CSMA/CA protocol in terms of 95th percentile



(a) Performance comparison (N=5, M=10) (b) Performance comparison (N=10, M=5) (c) Delay performance with different N_h

Fig. 12: The performance comparison under the collision model in heterogeneous networks with fixed $\hat{\lambda} = 0.5$.



(a) Performance comparison (N=5, M=10) (b) Performance comparison (N=10, M=5) (c) Delay performance with different N_h

Fig. 13: The performance comparison under the capture model in heterogeneous networks with fixed $\hat{\lambda} = 0.5$.

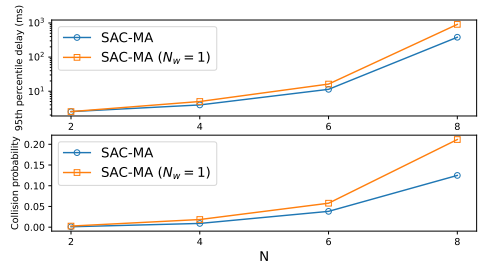


Fig. 14: The performance comparison between SAC-MA algorithm and its variant "SAC-MA ($N_w = 1$)".

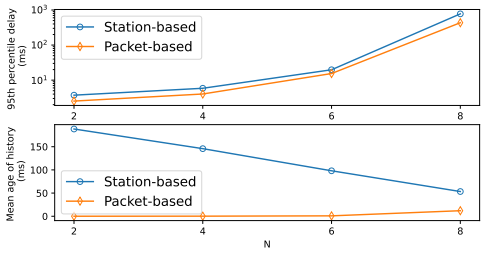


Fig. 15: The performance comparison between packet-based and station-based agent mechanisms.

delay, saturation throughput, and packet drop rate. On average across different traffic loads, SAC-MA reduces delay by approximately 27.9% and 56.5% compared to CSMA/CA in the collision and capture model, respectively. In dynamic environments, SAC-MA adapts rapidly to changing conditions including network sizes and traffic loads. Furthermore, simulation results confirm that SAC-MA can coexist harmoniously with legacy CSMA/CA stations, satisfying the 3GPP fairness criterion. Finally, ablation studies validate the effectiveness of the proposed mechanisms.

For future work, we plan to extend our algorithm to support multi-link operation, enabling devices to transmit over multiple frequency bands simultaneously. Additionally, we aim to explore traffic differentiation to support diverse quality of service requirements and learn effective traffic allocation across different links.

ACKNOWLEDGMENT

Any opinions, findings and conclusions or recommendations expressed in this material are those of the author(s) and do not reflect the views of National Research Foundation, Singapore.

REFERENCES

[1] E. Bastug, M. Bennis, M. Médard, and M. Debbah, "Toward interconnected virtual reality: Opportunities, challenges, and enablers," *IEEE Commun. Mag.*, vol. 55, no. 6, pp. 110–117, 2017.

[2] H. Huang, A. V. Savkin, M. Ding, and C. Huang, "Mobile robots in wireless sensor networks: A survey on tasks," *Comput. Netw.*, vol. 148, pp. 1–19, 2019.

[3] Z. Guo, Z. Chen, P. Liu, J. Luo, X. Yang, and X. Sun, "Multi-agent reinforcement learning-based distributed channel access for next generation wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 40, no. 5, pp. 1587–1599, 2022.
[4] "IEEE P802.11bn Project Authorization Request (PAR)," https://mentor.ieee.org/802.11/dcn/23/1123-0480-03-0uhr-uhr-proposed-par.pdf, 2023, accessed: 2024-02-06.
[5] Y. Xu, Z. Musgrave, B. Noble, and M. Bailey, "Bobtail: Avoiding long tails in the cloud," in *Proc. USENIX Symp. Netw. Syst. Des. Implementation*, 2013, pp. 329–341.
[6] W. Wydmański and S. Szott, "Contention window optimization in ieee 802.11 ax networks with deep reinforcement learning," in *Proc. IEEE Wireless Commun. Netw. Conf.*, 2021, pp. 1–6.
[7] S. Moon, S. Ahn, K. Son, J. Park, and Y. Yi, "Neuro-dcf: Design of wireless mac via multi-agent reinforcement learning approach," in *Proc. Mobile and Ad Hoc Netw. and Comput.*, 2021, pp. 141–150.
[8] C.-H. Ke and L. Astuti, "Applying multi-agent deep reinforcement learning for contention window optimization to enhance wireless network performance," *ICT Express*, vol. 9, no. 5, pp. 776–782, 2023.
[9] F. Frommel, G. Capdehourat, and F. Larroca, "Reinforcement learning based coexistence in mixed 802.11 ax and legacy wlans," in *Proc. IEEE Wireless Commun. Netw. Conf.*, 2023, pp. 1–6.
[10] A. Kumar, G. Verma, C. Rao, A. Swami, and S. Segarra, "Adaptive contention window design using deep q-learning," in *Proc. IEEE Int. Conf. Acoust., Speech and Signal Process.*, 2021, pp. 4950–4954.
[11] S. J. Sheila de Cássia, M. A. Ouameur, and F. A. P. de Figueiredo, "Reinforcement learning-based wi-fi contention window optimization," *J. Comm. Inf. Syst.*, vol. 38, no. 1, 2023.
[12] C. Lee, S. Park, and T. Cheong, "Dynamic-persistent csma: A reinforcement learning approach for multi-user channel access," *IEEE Access*, 2024.
[13] R. Yan, Z. Guo, P. Liu, Q. Lan, X.-P. Zhang, and Y. Dong, "Multi-agent reinforcement learning based channel access optimization for ieee 802.11 bn," *IEEE Trans. Green Commun. Netw.*, 2024.
[14] X. Du, X. Fang, R. He, L. Yan, L. Lu, and C. Luo, "Federated deep reinforcement learning-based intelligent channel access in dense wi-fi deployments," 2024, *arXiv:2409.01004*.
[15] Y. Shao, Y. Cai, T. Wang, Z. Guo, P. Liu, J. Luo, and D. Gündüz, "Learning-based autonomous channel access in the presence of hidden terminals," *IEEE Trans. Mobile Comput.*, vol. 23, no. 5, pp. 3680–3695, 2023.
[16] Y. Yu, S. C. Liew, and T. Wang, "Multi-agent deep reinforcement learning multiple access for heterogeneous wireless networks with imperfect channels," *IEEE Trans. Mobile Comput.*, vol. 21, no. 10, pp. 3718–3730, 2021.
[17] X. Ye, Y. Yu, and L. Fu, "Multi-channel opportunistic access for heterogeneous networks based on deep reinforcement learning," *IEEE Trans. Wireless Commun.*, vol. 21, no. 2, pp. 794–807, 2021.
[18] M. Han, X. Sun, W. Zhan, Y. Gao, and Y. Jiang, "Multi-agent reinforcement learning based uplink ofdma for ieee 802.11 ax networks," *IEEE Trans. Wireless Commun.*, vol. 23, no. 8, pp. 8868–8882, 2024.
[19] Z. Chen, X. Sun, Y. Jin, and F. Wang, "Multi-task reinforcement learning-based multiple access for dynamic wireless networks," to appear in *IEEE Trans. Mobile Comput.*, 2025.
[20] M. Han, Z. Chen, and X. Sun, "Multiple access via curriculum multi-task happo based in dynamic heterogeneous wireless network," to appear in *IEEE Internet Things J.*, 2024.
[21] L. Zhang, H. Yin, Z. Zhou, S. Roy, and Y. Sun, "Enhancing wifi multiple access performance with federated deep reinforcement learning," in *Proc. IEEE Veh. Technol. Conf.* IEEE, 2020, pp. 1–6.
[22] J. Xiao, Z. Chen, X. Sun, W. Zhan, X. Wang, and X. Chen, "Online multi-agent reinforcement learning for multiple access in wireless networks," *IEEE Commun. Lett.*, vol. 27, no. 12, pp. 3250–3254, 2023.
[23] Z. Chen and X. Sun, "Scalable multi-agent reinforcement learning-based distributed channel access," in *Proc. IEEE Int. Conf. Commun.*, 2023, pp. 453–458.
[24] J. Yu, L. Liang, Z. Guo, and S. Jin, "Heterogeneous multi-agent reinforcement learning for channel access in wlans," in *Proc. IEEE Wireless Commun. Netw. Conf.*, 2025, pp. 1–6.
[25] M. S. Oh, Z. Zhang, F. Hairi, A. Velasquez, and J. Liu, "Consensus-based decentralized multi-agent reinforcement learning for random access network optimization," 2025, *arXiv:2508.07001*.
[26] Z. Li, Y. Zhao, and J. Lee, "Multi-agent reinforcement learning for a distributed multi-channel access game," *ICT Express*, 2025.
[27] X. Zhao and L. Dai, "Throughput-optimal random access: A queueing-theoretical analysis for learning-based access design," 2025, *arXiv:1910.07207*.
[28] M. Han, Z. Chen, and X. Sun, "Bandit-based multiple access approach for multi-link operation in heterogeneous dynamic networks," *IEEE Open J. Commun. Soc*, 2025.
[29] Y. Hao, F. Li, C. Zhao, and S. Yang, "Delay-oriented scheduling in 5g downlink wireless networks based on reinforcement learning with partial observations," *IEEE/ACM Trans. Netw.*, vol. 31, no. 1, pp. 380–394, 2022.
[30] J. Kim, Y. Kim, S. Park, and J. Park, "Dynamic transmission and delay optimization random access for reduced power consumption," *IEEE Access*, vol. 12, pp. 55 033–55 050, 2024.
[31] J. Bae, J. Lee, and S. Chong, "Learning to schedule network resources throughput and delay optimally using q+-learning," *IEEE/ACM Trans. Netw.*, vol. 29, no. 2, pp. 750–763, 2021.
[32] I.-S. Comşa, S. Zhang, M. E. Aydin, P. Kuonen, Y. Lu, R. Trestian, and G. Ghinea, "Towards 5g: A reinforcement learning-based scheduling solution for data traffic management," *IEEE Trans. Netw. Service Manage.*, vol. 15, no. 4, pp. 1661–1675, 2018.
[33] Z. Gu, C. She, W. Hardjawana, S. Lumb, D. McKechnie, T. Essery, and B. Vucetic, "Knowledge-assisted deep reinforcement learning in 5g scheduler design: From theoretical framework to implementation," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 7, pp. 2014–2028, 2021.
[34] G. Bianchi, "Performance analysis of the ieee 802.11 distributed coordination function," *IEEE J. Sel. Areas Commun.*, vol. 18, no. 3, pp. 535–547, 2000.
[35] X. Sun and L. Dai, "To sense or not to sense: A comparative study of csma with aloha," *IEEE Trans. Commun.*, vol. 67, no. 11, pp. 7587–7603, 2019.
[36] V. Konda and J. Tsitsiklis, "Actor-critic algorithms," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 12, 1999.
[37] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.
[38] P. Christodoulou, "Soft actor-critic for discrete action settings," 2019, *arXiv:1910.07207*.
[39] "*IEEE Draft Standard for Information Technology–Telecommunications and Information Exchange Between Systems Local and Metropolitan Area Networks–Specific Requirements—Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications Amendment: Enhancements for Extremely High Throughput (EHT)*," standard IEEE 802.11be/D5.0, 2023.

Zhenyu Chen received the B.S. degree in Communication Engineering from School of Electronics and Communication Engineering, Shenzhen Campus of Sun Yat-sen University, Shenzhen, China in 2021. He is currently pursing for the Ph.D. degree in Electronic and Information Engineering with School of Electronics and Communication Engineering, Shenzhen Campus of Sun Yat-sen University, Shenzhen, China. His research interests include multiple access and reinforcement learning.

Xinghua Sun (M'13) received the B.S. degree from Nanjing University of Posts and Telecommunications (NJUPT), China, in 2008 and the Ph.D. degree from City University of Hong Kong (CityU), China, in 2013. In 2010, he was a visiting student with National Institute for Research in Digital Science and Technology (INRIA), France. In 2013, he was a postdoctoral fellow at CityU. From 2015 to 2016, he was a postdoctoral fellow at University of British Columbia, Canada. From July to Aug. 2019, he was a visiting scholar at Singapore University of Technology and Design, Singapore. From 2014 to 2018, he was an associate professor with NJUPT. Since 2018, he has been an associate professor with Sun Yat-sen University, Guangdong, China. His research interests are in the area of stochastic modeling of wireless networks and machine learning for next generation wireless communications and networks. He was a co-recipient of the Best Paper Award from the EAI IoTaaS in 2023 and the IEEE FCN in 2024.

Huadong Li received the B.S. degree in Electronic Information Engineering from the University of Science and Technology of China (USTC), China, in 2010, and the M.Phil. degree in Communication Engineering from The Chinese University of Hong Kong, Hong Kong, in 2012.

From 2012 to 2015, he was with Innofidei Inc. (Hong Kong), where he worked on physical-layer and higher-layer algorithm design for LTE chipsets and base stations, and contributed to the development and commercial deployment of LTE TDD systems. From 2015 to 2021, he was with Hytera Communications Corporation Limited, where he led the LTE algorithm team and was responsible for the design of radio resource management algorithms and protocol stack optimization. Since 2021, he has been serving as the General Manager of the Base Station Business Unit at Signalwing Corporation, Shenzhen, China, where he oversees product development and lifecycle management for 4G/5G base stations. In 2025, he was honored as a "Fenghuang Talent" by Bao'an District, Shenzhen Municipality.

Chenyuan Feng (S'16-M'21) received the B.E. degree in electrical and electronics engineering from the University of Electronic Science and Technology of China (UESTC), Chengdu, China, in 2016, and the Ph.D. degree in information system technology and design from Singapore University of Technology and Design (SUTD), Singapore, in 2021, respectively. Currently she is a research fellow at University of Exeter, U.K.. Her research interests include edge intelligence, AI for network and communication. Dr. Feng is a receipt of the 2021 IEEE ComComAp Best Paper Award, 2024 and 2025 IEEE ICCT Best Paper Award, and the 10th Anniversary ICCCS Best Paper Award. Dr. Feng was invited to deliver several tutorials and invited talk at International conferences in the area of machine learning for communication, such as IEEE PIMRC'2024, IEEE VCC'2024, IEEE VTC Spring 2025, etc. Dr. Feng also serves as an Associate Editor for the IEEE INTERNET OF THINGS JOURNAL and the IEEE OPEN JOURNAL OF THE COMMUNICATIONS SOCIETY. Dr. Feng is a Marie Skłodowska-Curie Scholar and 6G Star Young Scientist.

Xijun Wang (S'16-M'21) (M'12) received his B.S. degree (with high honors) in Communications Engineering from Xidian University in 2005, Xi'an, Shaanxi, China. He received the Ph.D. degree in Electronic Engineering from Tsinghua University in January 2012, Beijing, China. He was an Assistant Professor from 2012 to 2015 and an Associate Professor from 2015 to 2018 with the School of Telecommunications Engineering, Xidian University. From 2015 to 2016, he was a Research Fellow with the Information Systems Technology and Design Pillar, Singapore University of Technology and Design. He is currently an Associate Professor with Sun Yat-sen University. He received Best Paper Awards at IEEE ICCC 2013, FCN 2024, and ICCCS 2025. He was recognized as an Exemplary Reviewer of the IEEE WIRELESS COMMUNICATIONS LETTERS in 2014, 2021, and 2024. His current research interests include age of information, sematic communications, and distributed machine learning.

Qiaofeng Xue received the M.S. degree in Communication and Information Systems from Xidian University, China. She has extensive experience in wireless communications and full-stack R&D in 4G/5G technologies, including terrestrial communications, satellite internet, and domestic technology innovation.

At Signalwing Corporation, Shenzhen, China, she led the development of 4G LTE product series and served as a subsystem lead for the 5G NR Low Earth Orbit (LEO) satellite internet project, coordinating efforts to overcome key technical challenges in integrated space-ground communications. She also spearheaded the development of 5G product series based on x86 and ARM architectures to support diverse hardware platforms.

Tony Q.S. Quek (S'98-M'08-SM'12-F'18) received the B.E. and M.E. degrees in electrical and electronics engineering from the Tokyo Institute of Technology in 1998 and 2000, respectively, and the Ph.D. degree in electrical engineering and computer science from the Massachusetts Institute of Technology in 2008. Currently, he is the Associate Provost (AI & Digital Innovation) and Cheng Tsang Man Chair Professor with Singapore University of Technology and Design (SUTD). He also serves as the Director of the Future Communications R&D Programme, and the ST Engineering Distinguished Professor. He is a co-founder of Silence Laboratories and NeuroRAN. His current research topics include wireless communications and networking, network intelligence, non-terrestrial networks, open radio access network, AI-RAN, and 6G.

Dr. Quek was honored with the 2008 Philip Yeo Prize for Outstanding Achievement in Research, the 2012 IEEE William R. Bennett Prize, the 2015 SUTD Outstanding Education Awards – Excellence in Research, the 2016 IEEE Signal Processing Society Young Author Best Paper Award, the 2017 CTTC Early Achievement Award, the 2017 IEEE ComSoc AP Outstanding Paper Award, the 2020 IEEE Communications Society Young Author Best Paper Award, the 2020 IEEE Stephen O. Rice Prize, the 2020 Nokia Visiting Professor, the 2022 IEEE Signal Processing Society Best Paper Award, the 2024 IIT Bombay International Award For Excellence in Research in Engineering and Technology, the IEEE Communications Society WTC Recognition Award 2024, and the Public Administration Medal (Bronze). He is an IEEE Fellow, a WWRF Fellow, an AIIA Fellow, and a Fellow of the Academy of Engineering Singapore.