Distributed Age-of-Information Scheduling with NOMA via Deep Reinforcement Learning

Congwei Zhang, Yifei Zou, *Member, IEEE,* Zuyuan Zhang, Dongxiao Yu, *Senior Member, IEEE,* Jorge Torres Gómez, *Senior Member, IEEE,* Tian Lan, *Senior Member, IEEE,* Falko Dressler, *Fellow, IEEE,* and Xiuzhen Cheng, *Fellow, IEEE*

Abstract—Many emerging applications in edge computing require processing of huge volumes of data generated by end devices, using the freshest available information. In this paper, we address the distributed optimization of multi-user long-term average Age-of-Information (AoI) objectives in edge networks that use NOMA transmission. This poses a challenge of non-convex online optimization, which in existing work often requires either decision making in a combinatorial space or a global view of entire network states. To overcome this challenge, we propose a reinforcement learning-based framework that adopts a novel hierarchical decomposition of decision making. Specifically, we propose three different types of distributed agents to learn with respect to efficiency of AoI scheduling, fairness of AoI scheduling, as well as a high-level policy balancing these potentially conflicting design objectives. Not only does the proposed decomposition improve learning performance due to disentanglement of different design objectives/rewards, but it also enables the algorithm to learn the best policy while also learning the explanations – as actions can be directly compared in terms of the design objectives. Our evaluations show that the proposed algorithm improves the long-term average AoI by $200\% - 300\%$ and 400% compared to prior works with NOMA and the optimal solution without NOMA, respectively.

Index Terms—Age-of-Information, NOMA, Deep Reinforcement Learning, Distributed Computing

1 Introduction

The rapid development of 5G/6G communication technologies in recent years has prompted an ongoing shift toward an edge computing paradigm. New techniques such as Non-Orthogonal Multiple Access (NOMA) [1–3] and Age-of-Information (AoI) minimization [4–9] are considered to enhance transmission efficiency and to guarantee information delivery freshness in edge computing systems, resulting in significant improvement in response times and enabling highly elastic edge services.

In usual, an edge network (e.g. [10, 11]) contains multiple edge devices deployed on base stations (BSs) and massive end devices connected to the BSs via wireless channels. When a large number of end devices connect to the BSs, their uplinks to the BSs may fail due to the heavy interference and collisions in the open-access wireless channel. The use of NOMA and AoI minimization techniques can help address the uplink scheduling problem [12, 13]. Specifically, the NOMA technique allows multiple uplinks with the same destination to be scheduled simultaneously even under physical interference constraints [1–3, 14], and the concept of AoI accurately depicts the freshness of information, giving priority to uplink scheduling [15–18].

- *C. Zhang, Y. Zou (Corresponding Author), D. Yu and X. Cheng are with Institute of Intelligent Computing, School of Computer Science and Technology, Shandong University, P.R. China. E-mail:xczhw@mail.sdu.edu.cn, yfzou@sdu.edu.cn, dxyu@sdu.edu.cn, xzcheng@sdu.edu.cn*
- *Z. Zhang and T. Lan are with the Department of Electrical and Computer Engineering, The George Washington University, Washington, DC, 20052, USA. E-mail: zuyuan.zhang@gwu.edu, tlan@gwu.edu*
- *J. T. Gómez and F. Dressler are with the School of Electrical Engineering and Computer Science, TU Berlin, Berlin, 10587, Germany. E-mail: {torres-gomez, dressler}@ccs-labs.org*

Manuscript received MM DD, YYYY; revised MM DD, YYYY.

This paper studies distributed optimization of multi-user long-term average AoI objectives with NOMA in edge networks. The time in edge networks is divided into discrete slots, each of which is a time unit for users to transmit a packet in the wireless channel. In each time slot, the users can decide to transmit or listen, to update their AoI on the edge side. The final objective is to minimize the average AoI of all users in a sufficiently long interval, with channel contention and signal interference as the constraints.

Despite recent progress on NOMA and AoI minimization, optimizing AoI with NOMA is still an open problem. Specifically, it requires to solve a challenging non-convex online optimization that involves a number of discrete transmission powers as variables, different objectives, and control knobs. Firstly, while NOMA can substantially boost network throughput and potentially lead to improved AoI performance, it also gives rise to a difficult contention resolution and interference control problem that is known to be non-convex [2, 3, 19] and thus does not have an amenable solution. Secondly, the optimization of multi-user AoI must address a tradeoff between efficiency and fairness [20, 21]. That is, through a joint optimization, we need to achieve two (potentially) conflicting objectives – maximizing network throughput for efficiency of AoI scheduling and ensuring fairness among AoI received by different BSs. Finally, the nature of edge computing mandates a distributed online optimization of these problems, adapting to dynamic network conditions. However, existing works either formulate AoI scheduling with NOMA as an optimization with contention/interference constraints that lead to a combinatorial optimization [1, 20, 21] or require a global view/information about the whole network [15–18, 22–24].

To this end, we propose a reinforcement-learning (RL)

based framework that develops a novel *hierarchical decomposition of decision making*, for minimizing multi-user long-term average AoI with NOMA as the final objective. In particular, we focus on the uplink scheduling in edge networks and consider AoI for delivering data collected by end devices to edge servers located at 5G/6G base stations. We need to emphasize that while existing works have considered RL-based approaches for various network optimization problems [15–18, 22–24], they either focus on average AoI scheduling with a single, centralized agent or fail to address the tradeoff between different design objectives in decision making. Our key idea in this paper is a hierarchical decomposition of decision-making for distributed optimization of multi-user average AoI objectives. Specifically, we model (i) each base station as an individual learning agent that acts upon only local information, and (ii) each design objective – i.e., AoI efficiency (i.e., network throughput) and AoI fairness – through a separate learning agent with corresponding reward. Then, a high-level agent/policy is introduced to explicitly balance different design objectives. Leveraging independent Deep Reinforcement learning, it enables a distributed, online optimization algorithm as multiple agents, representing different design objectives and BSs, interact, self-teach, and learn to improve the decision-making policies on the fly.

In contrast to prior works that simply leverage existing RL algorithms for network optimization, our hierarchical decomposition of decision making provides a refreshing perspective. It not only decomposes the optimization into meaningful reward types (i.e., design objectives) that are more suitable for learning and thus lead to more favorable policies, but also enables the model to learn the best policy while also learning the explanations. Compared with some other long-term optimization techniques (i.e. Lyapunov technique), our distributed RL approach does not rely on any prior knowledge or assumptions, has a balanced exploration-and-exploitation to find the stable strategy, and is adaptive to the mobile environment.

The key contributions of the paper are as follows:

- We propose a reinforcement learning-based framework for distributed optimization of multi-user long-term average AoI objective with NOMA, a challenging non-convex online optimization problem.
- Our solution adopts a novel hierarchical decomposition of decision making, leveraging three different types of agents to learn with respect to efficiency rewards, fairness rewards, and a high-level tradeoff policy, respectively.
- The decomposition not only leads to improved learning performance but also provides interpretability – as to understand the reasons why an action has an advantage (or disadvantage) over another, in terms of efficiency and fairness.
- The framework is evaluated using an edge network simulator with 100 edge and 800 end devices. By comparing with the previous works [12, 13] with NOMA and the optimal solution without NOMA, the efficiency of our work has an improvement of 200%-300% and 400% in terms of minimizing the long-term average AoI, respectively. The performance of our algorithm is close to the optimal solution with NOMA, obtained by brute force.

The remainder of the paper is organized as follows. Sec. 2 introduces the related work. Sec. 3 formulates our AoI scheduling problem, NOMA-SINR model and the system deployment. Our AoI scheduling algorithm via learning is covered in Sec. 4, and Sec. 5 presents the simulation results for evaluation. Concluding remarks are given in Sec. 6.

2 Related Work

Firstly proposed in [25–27], Age-of-Information has become a quite emerging concept to characterize the freshness of the information. Currently, a series of works on AoI optimization problems have been presented in wireless networks and edge computing domain, including [4–9] in single hop networks, [28–33] in multi-hop networks and [19–21, 34] directly in edge computing framework. In [4], based on the SINR model, the authors prove that minimizing the overall information age is NP-hard and an integer linear programming formulation is provided to compute the global optimal solution. In [5] and [6], the AoI optimization problem for uplinks to a base station is considered. In [30], the authors first characterize AoI as a convex function of link activation rates in a single hop network and then extend the above result to an optimal policy in multi-hop networks. The authors in [31] directly consider the problem of minimizing average and peak AoI in multi-hop networks, and present an optimal stationary scheduling policy. As for those works considering the AoI scheduling problem in the edge computing domain, the authors in [21] consider the problem that an edge server collects the information from a source node over a delay channel and disseminates the information to its destination. Three online scheduling policies are proposed when the constraints and targets of optimization vary. In [19], a non-convex average AoI minimization problem is studied with energy and time constraints, which relies on the frequency division multiple access technique to provide a reliable communication environment. Unlike the centralized scheduling schemes mentioned above, a distributed solution for AoI scheduling in multi-hop networks is provided in [29]. To reach a local optimal scheduling on AoI, the algorithm in [29] adopts the distributed convex optimization technique, which entails considerable overhead for the information singular.

Apart from the above works without using learning skills, [15–18, 22–24, 35, 36] are a series of works optimizing the AoI scheduling process with learning schemes from different views. Over a perfect channel, the authors in [15, 22] use the deep Q-network (DQN) method and Q-learning to learn the data arrival statistics in AoI scheduling problem, respectively. An extension of [15] is provided in [18], in which various reinforcement learning methods are extensively simulated. For the unreliable channels, the work in [23] formulates the AoI problem as a restless multi-armed bandit, and proposes a suboptimal whittle index policy to solve it. In [24], nodes exchange status with each other in wireless ad-hoc networks, and employ the policy gradients and DQN methods to minimize their AoI. In [18], a reinforcement learning approach is proposed to minimize the long-term average AoI at users

when the channel statistics are unknown. In [35], a DQN-based scheme is proposed to guide the unmanned aerial vehicle to wirelessly charge multiple ground nodes. So that the average AoI of those ground nodes can be minimized.

Even though a series of works with/without learning schemes have been proposed in the above works to optimize the AoI scheduling problem, only a few of them use NOMA to facilitate the AoI scheduling process. The work in [1, 14] adopts NOMA technology to minimize the average AoI of downlinks. In [1], an adaptive AoI-aware buffer-aided transmission scheme is proposed to adjust the transmission rate and power in NOMA technology and improve the averaged AoI performance. In [14], a heuristic adaptation of the driftplus-penalty approach from the Lyapunov framework is used to minimize the average AoI with energy constraint. Also, the works in [2, 3] shows how the NOMA facilitate the transmissions in the wireless network, even though they are not specifically designed for AoI scheduling problem. Note that the works in [1–3, 14] have the similar assumption that the power domain is divided into constant levels for the PDMA (Power Division Multiple Access) in NOMA. Consequently, the speedup on message transmissions by adopting NOMA is limited, i.e., also a constant.

Compared with previous works, we remove the constant division on the power levels in the conventional NOMA transmission, and use deep reinforcement learning to fully explore the optimal solution for AoI minimization with NOMA. Consequently, a more efficient AoI scheduling in our work significantly outperforms that of previous works.

3 Network Model and Problem Definition

3.1 Network and Communication Model

We consider an edge network with m base stations (BSs) and n users (i.e., end devices) randomly placed on a 2-dimensional Euclidean space. We assume that BSs within a given distance are connected by wired links, and the users always choose the closest BS to connect via a wireless channel. In the most extreme case, in which the distance from the user to multiple BSs was the same, the user would randomly select a BS to connect. The time for communications between users and BSs is divided into equal-length discrete rounds, each of which contains three slots. For each round, the slot 1 is for the connected BSs to exchange information with each other via wired links, the slot 2 is for the BSs to broadcast the control messages to their users, and slot 3 is for the users to upload their information packets to BSs through a wireless channel. Figure 1 is an illustration for the three-slot communications in one round. In this paper, we mainly focus on the uplinks in slot 3, in which the data packets from users are uploaded to the edge servers at BSs. Additional assumptions for communications between BSs in slot 1, and downlinks from BSs in slot 2 are given in the following. As mentioned above, due to a constructed backbone network, the communications between neighboring BSs in slot 1 are stable and always succeed. Besides, we assume that the BSs are relatively far away from each other or scheduled by a TDMA (Time Division Multiple Access) scheme so that their downlinks in slot 2 will not interfere with each other. In other words, when a BS broadcasts its message in slot 2, all users connected to the BS can receive it. To depict the mobility of end devices, we assume that the users can move with a given speed in arbitrary directions in our 2-dimensional Euclidean space. If a user finds that its closest BS changes due to its mobility, it will connect to the new closest BS in the following rounds.

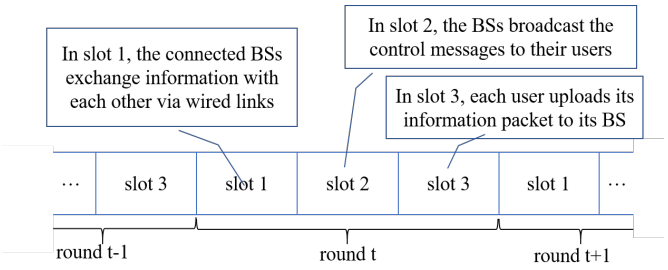


Fig. 1: Three-slot communications in one round.

In each slot of uplinks, multiple users may transmit signals to the BSs (i.e., receivers) and thus interfere with each other as the receivers try to decode their received signals. A signal will be decoded by the receiver when its strength is relatively larger than the strength of the interference plus the ambient noise. Different from traditional OMA (orthogonal multiple access) technologies, in which a receiver can at most decode one signal in each slot, NOMA makes it possible for a receiver to decode multiple signals in one slot. Specifically, by making full use of the SIC (Successive Interference Cancellation), the interference of a decoded signal can be removed from subsequent decoding processes of the remaining signals. To model NOMA in wireless communications with fading and interference cancellation, we consider the following NOMA-SINR equation:

$$SINR(u,v) = \frac{P_u / d(u,v)^\alpha}{\sum_{w \in S \setminus (S' \cup \{u\})} P_w / d(w,v)^\alpha + N}. \quad (1)$$

In the above equation, u and v are the transmitter and receiver, respectively; P_u is the power of the signal transmitted by u, $d(u,v)$ is the Euclidean distance between u and v. S is the set of synchronous transmitters. S' is the set of transmitters whose signals have been decoded by v. α is the path loss exponent and thus $d(u,v)^\alpha$ is the path loss between the transmitter u and the receiver v. β is a threshold determined by the hardware, and N is the ambient noise. In usual, $\alpha \in (2,6)$ and $\beta > 1$. In our SINR model, only when u transmits, v listens, and their SINR ratio $SINR(u,v)$ is larger than the threshold β, this signal from u can be decoded by the receiver v.

Power Level in NOMA Technique. According to Equation 1, for x users that connect to a base station b, by letting the i-th user (denoted by v_i) transmits with power $P_i = \beta(c\beta + c)^{i-1} \times d(v_i, b)^\alpha \times N$, in which constant $c \geq 1$ and $i = 1, 2, ...x$, the BS b can decode the messages from v_x, v_{x-1},..., v_1 one by one. Then, for the node transmitting with power P_i, we say it transmits with power level i in NOMA. From the above description, we can see that the energy consumption of a node exponentially increases when its power level gets larger. We use a positive integer $\hat{x}$ to denote the maximum power level in NOMA technique, which is in usual determined by the hardware and the energy budget of devices. Different from the previous works with $\hat{x} = 2$, a more general setting of $\hat{x} \geq 2$ is considered in our paper.

3.2 Definition and Problem Statement

To model the freshness of the information delivery from users to BSs, the definition of Age-of-Information similar to [37–40] is considered. Specifically, we assume that in slot 3 of each round t, each user v can choose one of the two actions after receiving the control message from its BS: (1) generate a data packet containing its most fresh information and transmit the generated data packet to its BS instantly, (2) do nothing. Let the AoI with respect to a user v be the time that elapsed since the generation of the freshest message transmitted from v to its BS, and let $\mathcal{A}_v(t)$ be the AoI of the user v at round t. According to the above assumption, once a data packet from v was received and successfully decoded by its BS at round $(t-1)$, $\mathcal{A}_v(t)$ drops to 1, and increases linearly in time otherwise. By setting $A_v(0)=1$ initially, and defining $\mathcal{E}_v(t-1)$ as the event that the freshest packet from v is received by its BS at round $(t-1)$, the $A_v(t)$ has the following formulation.

$$A_v(t)=\begin{cases}1 & \mathcal{E}_i(t-1) \text{ occurs,}\\ A_v(t-1)+1 & \text{otherwise,}\end{cases}$$

For a base station, it maintains all the AoI of its users.

Definition 1. **Average Age-of-Information.** For each user v, its average Age-of-Information (AoI) in an interval I that contains $|I|$ rounds is defined as $\tilde{A}_v(I)=\sum_{t=1}^{|I|}A_v(t)/|I|$. For the end-to-edge information system, its average AoI is the average value of $\tilde{A}_v(I)$ for all users, i.e., $\tilde{A}(I)=\sum_{v\in V}\tilde{A}_v(I)/n$, in which V and n are the set and number of users.

As has been discussed in [32], the AoI scheduling has a close relationship with the throughput of the network. In this paper, the throughput of each BS is also defined to help our algorithm design.

Definition 2. **Throughput.** Let tp_t^b be the throughput of the BS b in round t. If the BS b successfully receives packets from f users in the slot 3 of round t, $tp_t^b=f$.

Our Optimization Problem. Our objective is to minimize the average AoI of the end-to-edge information system in a sufficient long interval, which is also termed as multi-user long-term average AoI minimization problem in this paper. Note that our optimization problem is based on the NOMA transmission. Thus, in each slot 3 of a round, every user can choose to transmit or not, by its own will or according to the control message from its BS. If to transmit, it also has to determine the power level to satisfy the successive interference cancellation requirement in NOMA. Then, the BSs update the AoI of their users according to the Equation 1. Let π denote a stochastic policy. π is a random variable representing the probability of performing an action in a given state Thus, the final scheduling policy for our optimization problem can be denoted as sequences $\pi_v=\{\pi_v(1),\pi_v(2),...,\pi_v(|I|)\}$ for each node $v\in V$, where $\pi_v(t)\in\{0,1,2,...,\hat{x}\}$. $\pi_v(t)=i$ with integer $i=1,2,...,\hat{x}$ means the user v will transmit its freshnest data packet with power level i at round t. Additionally, we use $\pi_v(t)=0$ to denote the special case that v keeps silent in round t. Thus, the multi-user long-term average AoI minimization problem has the following formulation:

$$\min_{\pi}\tilde{A}(I)=\frac{1}{|I|n}\sum_{v\in V}\sum_{t=1}^{|I|}A_v(t) \text{ with } \pi=\cup_{v\in V}\pi_v, \quad (2)$$

in which I is a sufficient long interval and π is the scheduling policy for all nodes. From a given policy π and NOMA-SINR Equation 1, we know whether the data packet of a node v can be received by its BS at each round t of the interval I. Then, the AoI of v in interval I can be evolved. Our final objective is to find such a scheduling policy π to minimize the average AoI of all nodes in the interval I.

The most important notations and parameters are listed in the Table 2 for reference.

Acronyms	Definitions
BS	Base station
BF	Brute force
AoI	Age-of-Information
SIC	Successive Interference Cancellation
OMA	Orthogonal multiple access
NOMA	Non-orthogonal multiple access
PDMA	Power division multiple access
TDMA	Time Division Multiple Access
DeepRL	Deep reinforcement learning
Notations	**Definitions**
m	The number of BS
n	The number of users
t	Current round
$\|I\|$	A time interval
V_b	The set of users in BS b
N	Ambient noise determined by environment
$d(u,v)$	Euclidean distance between u and v
P_u	Transmission power of transmitter u
α,β	SINR parameters
$A_v(t)$	AoI of user v at round t
$\tilde{A}_v(I)$	The average AoI of user v in the interval I
$\tilde{A}(I)$	The average AoI of all users in the interval I
S	The set of synchronous transmitters
S'	The set of transmitters whose signals have been decoded
$\mathcal{E}_v(t-1)$	The event that packet from a transmitter v is received by its BS at round t
tp_t^b	The throughput of the BS b in round t
k_t^b	Highest power level in NOMA technique in round t for the users in BS b
$s_t^{b,X}$	State vector of the X-agent(X is one of T, F and B) in BS b at round t
$a_t^{b,X}$	Action vector of the X-agent(X is one of T, F and B) in BS b at round t
$r_t^{b,X}$	Reward vector of the X-agent(X is one of T, F and B) in BS b at round t
$J(\alpha)$	The temperature parameter of SAC-Discrete algorithm.
$\bar{H}$	A constant vector equal to the hyperparameter representing the target entropy of SAC-Discrete algorithm
$Q(a_t)$	The Q-value of action a_t
π_v	The policy of user v on average AoI minimization
γ	Attenuation factor in Q-learning.
θ	The parameter of the actor network.
ω	The parameter of the critic network.

TABLE 1: Table of acronyms and math notations.

4 Algorithm Description

4.1 Challenges and Solutions

In this paper, we propose a learning-based algorithm to solve the Age-of-Information scheduling problem with NOMA transmission. The proposed algorithm is a distributed one in terms of the BSs, as is illustrated in Figure 2(a). In other words, in each round t, according to the communications in slot 3 of round $(t-1)$ and slot 1 of round t, each BS knows

the information of its users and neighboring BSs, respectively. Then, each BS independently executes our learning-based algorithm to figure out the transmission policies for all its users and disseminate them by broadcasting in slot 2 of the current round. Then, in slot 3 of round t, each user v transmits or not according to its policy $\pi_v(t)$, and its AoI at the current round gets updated by its BS.

The objective of our algorithm is to minimize the long-term average AoI of all users. An intuitive learning-based approach for this optimization is to use the average AoI as the reward to guide the learning process. Whereas, in our final learning framework, two RL agents (T- and F-agents) are designed at each BS to optimize its efficiency and fairness of information scheduling, respectively. Then, a third RL agent (B-agent) leverages the actions generated by these two agents and aims to learn a high-level policy to make final scheduling decisions at the BSs. The total three agents on each BS constitute our distributed learning framework. In the following, we further explain why we need such a multi-agent setting in each BS.

Firstly, due to multi-hop connections between BSs, it is hard for each BS to obtain the real-time average AoI of all users in its learning process. Even though the local average AoI is accessible for each BS, [1] directly adopting it as the reward of learning misleads the BS on AoI minimization. Specifically, for each BS b, no matter which policies the other BSs take, choosing the highest power levels in NOMA for its uplinks makes its local average AoI more likely to be reduced in the next round. Because its uplinks with higher power levels are more likely to succeed, despite the interference from the users of other BSs. However, when all BSs choose the highest power levels for their users, the heavy interference with each other fails most of the uplinks and the average AoI of all users gets increased. Secondly, the average AoI may not serve as an accurate reward signal, because the average AoI is a time accumulative metric while the states, rewards, and actions in learning process are mainly roundly based, which reduces the accuracy of learning.

To avoid such disadvantages, it is important for BS to appropriately contend the multi-access channel for their users to minimize the local average AoI and meanwhile control their interference to the link schedulings proceed by other BS. Since the contention and interference in the wireless channel cannot be intuitionistically observed in AoI minimization problem, an alternative approach proposed in this paper is to optimize the efficiency and fairness simultaneously in information scheduling. First, we can show that maximizing throughput and minimizing average AoI are two equivalent problems for a base station. Therefore, each BS tries to maximize its throughput to minimize its local average AoI.However, maximizing local AoI alone can create a prisoner's dilemma between base stations, leading to heavy competition. Hence, the fairness with its neighbors in terms of minimizing the local average AoI should be maintained. The detailed proof is in the appendix. In other words, a BS should firstly try to maximize its throughput. Thus, the uplinks from its users with the largest ages can be efficiently scheduled, and the local average AoI at the next round can be minimized. Meanwhile, when a BS finds that its average AoI is much smaller than that of its neighbors, it should choose the lower power levels in NOMA to decrease its interference to its neighboring BS, so that more uplinks can be scheduled by its neighbors and the global average AoI becomes smaller.

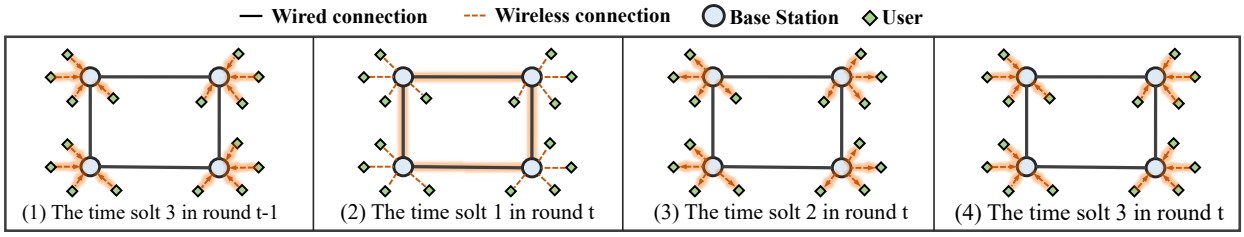
Both of the optimizations on efficiency and fairness are important for the global average AoI minimization problem. As discussed above, each BS only maximizing its own throughput results in a local optimal and energy consuming policy. If the fairness is optimized without considering the efficiency, a "lazy" policy cannot be avoided, in which all users choose not to transmit and their AoI equally increases. However, achieving both efficiency and fairness in a single-agent system is nearly impossible, due to the inherent conflict between these two objectives. For example, a strategy for a BS to maximize its efficiency would allocate higher power levels to its users, potentially exacerbating the unfairness due to its interfere to the uplink schedulings of other BSs. A solution to avoid this contradiction is to use two different learning agents (denoted by T-agent and F-agent) to address efficiency and fairness separately, and leverage another agent (denoted by B-agent) at each BS to integrate the actions generated by T-agent and F-agent, in order to obtain a joint action for the AoI scheduling problem. This B-agent optimizes the weights for combining different actions to balance efficiency and fairness, which allows direct interpretation of our learned outcomes.

In summary, the real-time global average AoI is not accessible for BSs in the multi-hop edge network, and optimizing the local average AoI misleads the learning process, due to the time-accumulative nature of AoI. Maximizing the throughput of BS in each round is equivalent to minimizing the MLA-AoI in a single-hop wireless network. Whereas, in a multi-hop wireless network, each BS maximizing its throughput results in a Prisoner Dilemma. Thus, fairness is considered as one of the objectives, to avoid the Prisoner Dilemma. Our approach in this paper is to optimize the efficiency and fairness of BSs in each round during the AoI scheduling process. Then, considering the branches on optimizing these two problems, two agents are designed on each BS for the efficiency and fairness problems, respectively. Finally, to well integrate the branched actions from the two agents, a reinforcement learning agent is used. Additionally, the learning states and rewards designed in our work do not rely on the number of users and network topology. Thus, compared with the existing work [12] whose states are closely determined by the number of users, our learning scheme is more concise and stable, especially when the position of users changes in mobile networks.

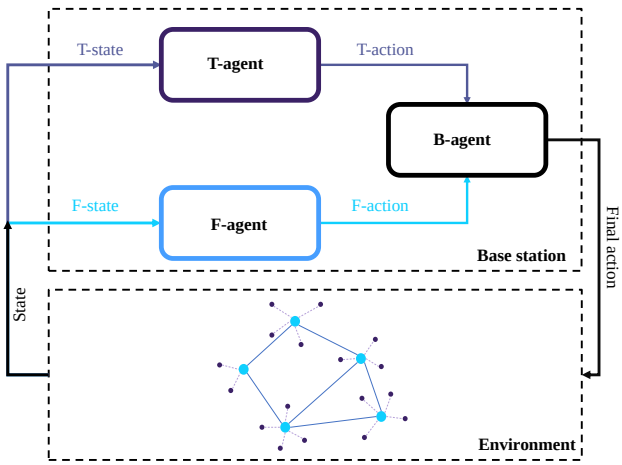
4.2 Distributed Learning Framework

As mentioned above, we leverage three different types of learning agents to optimize the efficiency and fairness objectives and to balance them in a joint optimization. Specifically, for each BS, we define T-agent for throughput optimization, F-agent for fairness optimization, and B-agent for learning a high-level policy for balancing the two design objectives, respectively. T-agent in each BS aims to maximize its throughput in each round relying only on local information, thus enabling a distributed solution; F-agent

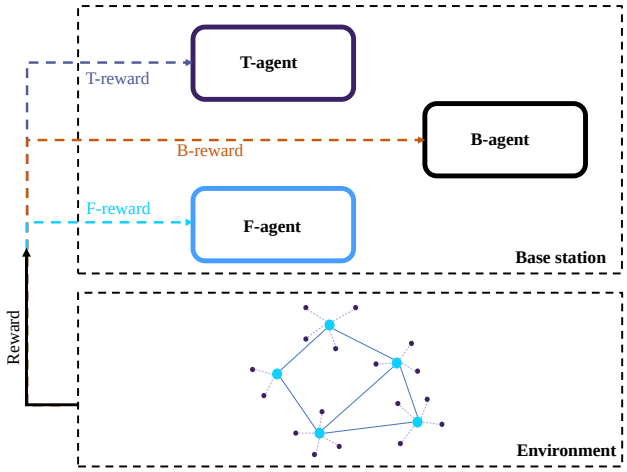
1. The local average AoI for a BS means the average AoI of the users connected to the BS.



(a) 3-slots communications in each round.



(b) Principles of the DeepRL algorithm evaluating process.



(c) Principles of the DeepRL algorithm training process.

Fig. 2: Distributed deep reinforcement learning approach for AoI optimization.

adjusts the average AoI of its own users and aims to balance the average AoI of users connected to the neighboring BSs. Finally, B-agent decides how to balance the two design objectives by choosing the actions generated by T-Agent and F-Agent. And the action from B-agent is directly executed by the BS. This hierarchical decomposition of decision making enables the model to learn the best policy while also to learn the explanations. Since B-agent adopts a high-level policy discriminating the actions generated by T-Agent and F-Agent, it allows actions to be directly compared in terms of tradeoffs between the two design objectives. It helps understand the reasons why an action of BS b at round t has an advantage (or disadvantage) over another, with respect to efficiency and fairness in optimizing multi-user AoI objectives with NOMA.

Deep reinforcement learning (RL) scheme adopted in our agents is a combination of reinforcement learning and deep neural networks (DNNs), which is more powerful than RL in handling complex tasks. Different from the supervised learning schemes with external knowledge guidance, Deep RL agents learn their behaviours by interacting with the environment. In each iterative step, the Deep RL agents observe the current state of the environment and make corresponding decisions/actions. The environment then evolves from the current state to a new one and returns a reward to the agents, which is a feedback about the quality of the agents' actions. The final goal of the agents is to learn a strategy, which is a DNN that maps states to all action rewards. To find a satisfactory strategy, Deep RL takes an exploration-exploitation-based method. A Deep RL agent can empirically select the action with the largest reward in the current state, which is called action exploitation. Also, it can try a new action for a potentially higher reward, which is called action exploration. A good balance between exploitation and exploration helps the agent "understand" the environment well and learn an optimized policy through enough iterations.

Figure 2 is presented to illustate our framework. Specifically, in Fig.2 (a), the first subgraph represents that at time slot 3 of the previous round, the BSs receive signals from their users and update the relevant information according to the received packets. The second subgraph shows that at time slot 1 of the current round, the wired-connected BSs exchange the required information with each other to obtain the average AoI of their neighboring BSs. The third subgraph reveals time slot 2 of the current round. BSs use Deep RL agents to obtain their actions. Afterward, the BSs broadcast the commands to their users, about which users should send signals with which power. The final subfigure shows the time slot 3 of the current round, users send signals to BSs or not based on the messages from BSs. Then, BSs update the AoI information as the reward. In In Fig.2 (b), each BS uses its T-agent and F-agent to obtain its T-action and F-action, according to its state from the environment. Then, the T-action and F-action will be combined into the final action by B-agent. The BS delivers the final action to its users at the end of the time slot 2. In Fig.2 (c), the users execute the final action and T-/F-/B-agent get a reward from the environment in slot 3. The rewards fed from the environment are forwarded to T-/F-/B-agent to update itself.

4.3 Design of T-agents

Aiming at maximizing the throughput of BSs, the state space, action space and reward in T-agent are designed as follows.

State Space. As the input of an agent, the state is the most direct way for the agent to know the network environment in each round, and is the most important basis for it to select the appropriate actions. Besides, the action to be taken by the agent in the coming round is also partially determined by the state. Thus, a well designed state space not only should include all the useful information for the agent to take the appropriate action, but also should be as brief as possible. Obviously, if some important features of the network environment are missing, it is very likely for the agent to get confused or be misled, which causes the failure of convergence in a learning process. On the other hand, when too much irrelevant information is contained in the state space, it will be hard for the agent to find the hidden relationship between the states and the optimal actions.

Formally, we define $s_t^{b,T}$ as the state vector of the T-agent in BS b at iteration step t, given as

$$s_t^{b,T} = [k_{t-1}^b, tp_{t-1}^b]. \tag{3}$$

In the above Equation, k_{t-1}^b is the highest power level in NOMA technique in round $t-1$ for the users in BS b; and tp_{t-1}^b is the throughput of BS b in round $(t-1)$. k_{t-1}^b indicates the contention level of BS b for the wireless channel, and state vector $s_t^{b,T}$ records the throughput of BS b under such a contention level in last round.

Action Space. The agent interacts with the environment by choosing an action. For each agent, its action in each round is mainly determined by its received state in current round and its learning experience so far. In this paper, we define $a_t^{b,T}$ as the action vector of the T-agent in BS b at round t. The action vector can be expressed as Equation 4

$$a_t^{b,T} = [k_t^{b,T}], \tag{4}$$

in which $k_t^{b,T} \leq \hat{x}$ is the highest power level in NOMA technique for users connected to BS b at round t.

Reward. After an action was chosen by the agent, a reward will be returned from the environment. Ideally, when the agent makes a good action, the environment should return a positive reward; on the other hand, when a bad decision was chosen by the agent, the agent should receive a negative reward. In this way, agents can be motivated to execute those good actions to maximize their rewards, which makes the learning scheme work. Note that the reward values are computed through a reward function based on current network status. Thus, the reward function on each agent should be carefully designed to make sure those good actions with maximum rewards are also the optimal solutions for the real problem we want to solve.

Aiming at maximizing its throughput, the throughput of a BS can be directly set as the reward of its T-Agent. A formal definition is given in Equation 5, in which $r_t^{b,T}$ is the reward received by the agent T of BS b on round t.

$$r_t^{b,T} = tp_t^b \tag{5}$$

4.4 Design of F-agents

The goal of F-agent on each BS b is to average the AoI of users which belongs to BS b and its neighbors. The designs of its state space, action space and reward are given below.

State Space. Different from the state space in T-agent, in which only b's own information is used, the state space in F-agent not only includes the age information of users within BS b, but also contains the age information of the users which belong to b's neighbors. Equation 6 is a formal definition for the state vector in F-agent, in which $s_t^{b,F}$ denotes the state of F-agent on BS b in round t, V_b is the set of users connected to BS b. Set U_b includes all the neighboring BSs of b.

$$s_t^{b,F} = \left[\frac{\sum_{v\in V_b} A_v(t)}{|V_b|}, \frac{\sum_{b'\in U_b}\sum_{v\in V_{b'}} A_v(t)}{\sum_{b'\in U_b}|V_{b'}|}\right] \tag{6}$$

Action Space. Similar with the action space of T-Agent, the action space of F-agent is also the highest power level in NOMA technique. In Equation 7, $a_t^{b,F}$ is an action made by F-agent at BS b in round t with $k_t^{b,F} \leq \hat{x}$.

$$a_t^{b,F} = [k_t^{b,F}] \tag{7}$$

Reward. The reward of the F-agent is about the difference between the average age of users within BS b and the average age of users within b's neighboring BSs. Consider that larger the difference, farther away an action is from the goal of the F-agent. Thus, the reward in F-agent is set as the inverse of the difference. A formal definition is given in the following Equation 8.

$$r_t^{b,F} = \frac{1}{\left|\frac{\sum_{v\in V_b} A_v(t)}{|V_b|} - \frac{\sum_{b'\in U_b}\sum_{v\in V_{b'}} A_v(t)}{\sum_{b'\in U_b}|V_{b'}|}\right|} \tag{8}$$

4.5 Design of B-agents

So far, we have presented the states, actions, and rewards of T-Agent and F-agent, but left the integration part unsolved. Note that the optimal point on integrating the actions from T-agent and F-agent varies with many factors, such as the network topology. In this part, we design an independent agent to learn the optimal integration ratio of the actions from T-Agent and F-agent.

Action Space. The action of B-agent is the integration ratio of the actions from F-agent and T-Agent. Define $a_t^{b,B}$ as the action of B-Agent on BS b in round t, and a_t^b as the final action of BS b in round t. The relationship between a_t^b, $a_t^{b,T}$, $a_t^{b,F}$ and $a_t^{b,B}$is given as

$$a_t^b = \lfloor a_t^{b,B} \times a_t^{b,T} + (1 - a_t^{b,B}) \times a_t^{b,F} + 0.5 \rfloor \tag{9}$$

Since the action $a_t^{b,T}$ of T-agent and action $a_t^{b,F}$ of F-agent are all the highest power level in NOMA, the final action a_t^b of the BS b at round t will also be the highest power level in NOMA. To ensure that the final action is an integer, a_t^b is rounded from $a_t^{b,B} \times a_t^{b,T} + (1 - a_t^{b,B})$ in Equation 9. Specifically, in round t, the BS will sort its users in descending order in terms of their AoI, and let the i-th user transmit with the power level $\max\{a_t^b - i + 1, 0\}$. In other words, the power level a_t^b will be allocated to the user with the largest AoI, and the user with the second largest

AoI will transmit with the power level of $a_t^b - 1$. When all the power levels $\{1, 2, ..., a_t^b\}$ have been assigned, the other users will have the power level 0, i.e., listening. For example, if there are 6 users numbered $1-6$ under the current BS, with their AoI as $\{3, 1, 3, 2, 2, 4\}$. By executing our DeepRL algorithm, the BS gets a final action of 3. That is to say, in the current round, this BS gets 3 quotas of communications using the power levels $1, 2$, and 3, respectively. Then, the action received by the users will be: user #6 sends signal with the power level 3; user #1 sends signal with the power level 2; and user #4 sends signal with the power level 1. The users #2, #3, and #5 do nothing. These policies from the BSs will be disseminated to their users in slot 2. In slot 3, the users will transmit according to the policies received from their BSs.

Reward. Here, we set the average AoI of users within BS b and its neighbors as the reward of B-agent on BS b. Obviously, the smaller the reward value we have, the better the integration ratio we selected in B-agent. $r_t^{b,B}$ is the reward of B-agent on BS b in round t, as is shown in Equation 10.

$$r_t^{m,B} = -\frac{\sum_{b' \in U_b \cup \{b\}} \sum_{v \in V_{b'}} A_v(t)}{\sum_{b' \in U_b \cup \{b\}} |V_{b'}|} \tag{10}$$

It is necessary to say, that the reward of B-agent is very close to the original problem (2). Specifically, the optimization objective in the original problem (2) is to minimize the global average AoI. To ensure the learning objective of B-agent is the same as that of the original problem (2), the best solution is to directly set the global average AoI as the reward. Whereas, such global information is time-consuming to obtain in a distributed system, which reduces the efficiency of our learning process. So, we settle for the second best by using the local AoI as the reward of the B-agent.

4.6 Summary of the Training Algorithm

T-agent and F-agent in our solution are implemented by soft actor-critic (SAC)[41] algorithms based on Deep RL, to find some *good* solutions for the throughput of BS and fairness of users. The B-agent is constructed by an RL algorithm to combine the actions of T-agent and F-agent. Thus, an efficient AoI scheduling can be achieved.

As one of the efficient RL algorithms, SAC uses the entropy-regularized formalism to augment exploration [41]. This approach is based on an AC (actor-critic) framework that specifies the stochastic policy and soft Q-function separately. It attempts to find a stochastic policy that maximizes the expected cumulative reward while taking as many different actions as possible. But the sample SAC algorithm is used for continuous action settings. So in this paper, we use a discrete variant of SAC learning called SAC for discrete (SAC-Discrete) to handle the discrete settings by fitting the probability of the action [42].

In the policy evaluation step of soft policy iteration, the SAC-Discrete algorithm aims to compute the value of a policy $\hat{\pi}$ according to the maximum entropy objective.[2] Different from the SAC algorithm, the SAC-Discrete algorithm can make the soft Q-function output the Q-value of each possible action rather than simply the action provided as an input, i.e., the Q-function moves from $Q : S \times A \to R$ to $Q : S \to R^{|A|}$. The policy is defined as $\hat{\pi} : S \to [0, 1]^{|A|}$ to output the action distribution. A softmax function is adopted in the final layer of the policy to ensure that a valid probability distribution for all actions can be output.

Because the action set is discrete, we can calculate the expectation for actions directly. The soft state-value calculation equation is defined as:

$$V(s_t) := \hat{\pi}(s_t)^T \left[Q(s_t) - \alpha \log\left(\hat{\pi}(s_t)\right) \right]. \tag{11}$$

In the above definition, $V(s_t)$ refers to the state-value at state s_t, which represents the expected future reward obtained by the following policy $\hat{\pi}$ from state s_t onwards. The Q-value at the state s_t is denoted by $Q(s_t)$. The discount factor is represented by α, which balances the importance of immediate and future rewards in the Q-function. And the policy $\hat{\pi}$ determines the probabilities of different actions to be taken in a given state.

The SAC-Discrete algorithm also provides an optional way of learning the temperature parameter $J(\alpha)$ so that we do not need to set it as a hyperparameter [42]. The formulation of $J(\alpha)$ is given in the following,

$$J(\alpha) = \hat{\pi}_t(s_t)^T \left[-\alpha \left(\log\left(\hat{\pi}_t(s_t)\right) + \bar{H} \right) \right], \tag{12}$$

in which $\bar{H}$ is a constant vector equal to the hyperparameter representing the target entropy.

Similar to many RL algorithms, the SAC-Discrete algorithm adopted in our algorithm also uses the double network to avoid overestimating state-action values. Specifically, we use Figure 3 and Figure 4 to show the update strategy for critic and actor in SAC, respectively. In those two figures, the quadruple (s, a, r, s') means the state in this round, the action made by the agent, the reward in this round, and the state in the next round. For the actor, the gradient descent with policy loss is used to update the policy $\hat{\pi}$ in the actor-network. For the critic network, two critics are designed for overestimation avoidance.

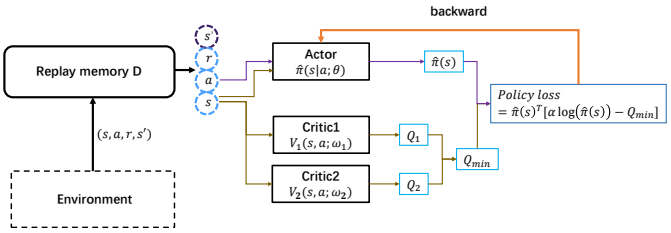


Fig. 3: Update dialog of SAC's actor network

Specifically, Figure 3 shows the updating of actor networks. The actor network is set to get the policy of the algorithm and two critic networks assist actor network updates. When a quadruple (s, a, r, s') is given, the actor network calculates the policy $\hat{\pi}(s)$ in the state s. Then the two Q-value Q_1 and Q_2 of (s, a) were obtained through the two critic networks, respectively. The smaller Q-value is used to update the actor-network by backward of the policy loss function to prevent the overestimation phenomenon in the reinforcement learning process.

The update process of critic networks is shown in Figure 4. The two critic networks are used to get the Q-value of a

2. Here, we use $\hat{\pi}$ to denote a policy in the SAC-Discrete algorithm, which is distinguished from $\pi_v(t)$, the policy of a user v at round t.

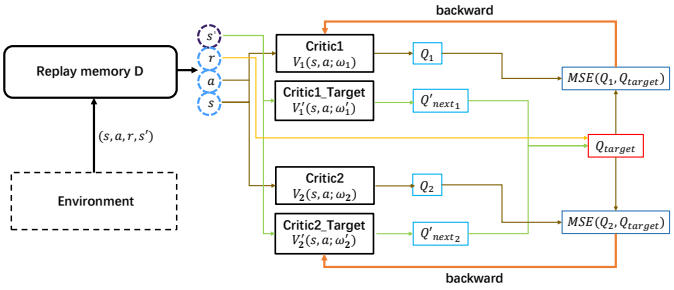


Fig. 4: Update dialog of SAC's critic network

specific state-action pair. Two target critic networks are used to correct the Q-value calculated by the critical network to assist in updating. The two critic-target networks get Q'_{next_1} and Q'_{next_2} by the forward process. The smaller one is chosen to calculate the MSE loss function and do the backward process for the critic network updating to prevent the overestimation. In addition, in every round, the target network will be soft updated according to the critic network.

As for the reinforcement learning algorithm deployed on the B-agent, no input is required. Due to the small state and action space, we adopt a tabular RL algorithm, in which a Q-table array is used to store the Q-value for every discrete action. An ϵ-greedy algorithm is employed to determine the action taken by the B-agent in each round. With probability $\epsilon < 1$, the action with the largest Q-value is selected. And with the other probability, an action is randomly chosen from the Q-table. The selected action will serve as the rate for balancing the outputs of the T-agent and F-agent. When the reward is returned, the B-agent updates its Q-table through the Bellman Equation shown in Equation 13, in which a_t is the action chosen by the agent in round t, $Q(a_t)$ is the Q-value of action a_t, r_t is the reward in round t, and α, γ are the learning parameters of RL algorithm.

$$Q(a_t) \leftarrow Q(a_t) + \alpha \left[r_t + \gamma \max_a Q(a) - Q(a_t) \right] \tag{13}$$

5 Numerical Evaluations

In this section, numerical simulations are conducted to demonstrate the efficiency of our proposed algorithm on AoI scheduling with various network sizes and mobility of users. Firstly, the minimum/average/maximum AoI w.r.t. all users at each round is observed, which directly reflects the performance of our algorithm. Secondly, to minimize the average AoI, three agents with different objectives have been specifically designed in our distributed learning-based algorithm. The T-agent is designed for efficiency of information scheduling, i.e., to maximize the throughput of BSs. The F-agent is adopted for fairness of users, i.e., to let the users with larger AoI more likely to be scheduled. The B-agent is used to fuse the policies from T- and F-agents. To verify this idea in our algorithm design, the minimum/average/maximum throughput of BSs are also observed. Additionally, by comparing the gaps between the minimum/average/maximum values of AoI and throughput, the fairness can be verified. Besides, to further investigate the execution of T-, F-, and B-agents, the loss functions of T-, F-agents and weight parameters of B-agents are observed. Finally, the comparisons with previous works are conducted to show the advantage of our algorithm on AoI scheduling. In our simulation, we have the number of base stations increased from 2 to 100, and the static and dynamic modes are designed for at most 800 users, which simulate the various network sizes and mobility of users.

5.1 Simulation Setup

We simulate our edge computing system in a 2-dimensional Euclidean space, with the number of edges and end devices varying within $[2, 100]$ and $[16, 800]$, respectively. In general, our simulated edge network is based on hexagonal cells, each of which contains 1 base station and 8 mobile users. As illustrated in Figure, the base station is in the central point of the hexagonal cell, and 8 users are randomly and uniformly deployed within the hexagonal area. Each side of the hexagonal cell has a length of 3.5 km. According to the numerical results observed in our simulation, implementing 8 users for each BS is enough to let the BS fully busy on AoI scheduling and test the performance of our algorithm on AoI scheduling, throughput and fairness. Even though in some realistic scenarios, there are more than 8 user devices within the communicaiton range of a BS. By clustering them into groups, each of which contains 8 end devices, and adopting a TDMA technique, those scenarios can be simplified to our setting in this simulation. To simulate a multi-hop network with multiple edge devices, more hexagonal cells are added in our simulation according to the number of BSs, as is illustrated in Figure 5 . When those hexagonal cells are implemented, the base stations within a distance of 20 km of each other are connected by wired links; each user will connect with the nearest BS. As for the parameters α, β, c, N in our SINR-NOMA model, we have $\alpha = 3.0$, $\beta = 1.5$, $c = 1.5$ and the noise N normalized as 1.0. Additionally, the static and dynamic modes for users are considered in our simulation. In the static mode, the location of users will not change. While in the dynamic mode, the users randomly move in an arbitrary direction in our 2-dimentional space. The speed of a user in each round is a random variable from the interval $[0, 0.2]$ km/round. Once the nearest BS changes due to is movement, it will connect to the new nearest BS.

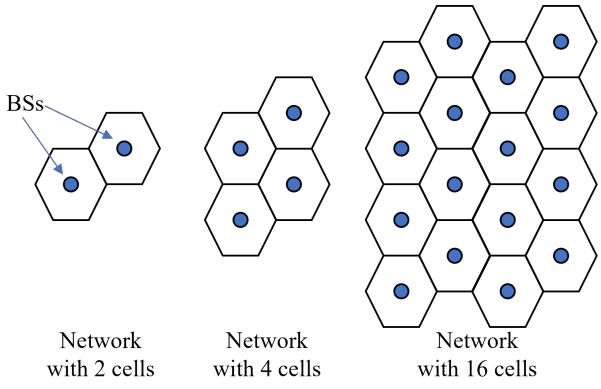


Fig. 5: By increasing the number of cells, the edge network with more BSs is simulated

After the implementation of network topology, our learning scheme on each edge automatically initializes. In each

round, the learning agents in each BS will decide how many users should send messages with what level of transmission power in current round. After that, all users follow the commands from their BSs to send messages with specific transmission power levels. After one round execution, each BS collects the parameters from its local environments, as the input for its learning agents. After receiving the necessary parameters, the learning agent updates its current state, and returns the new actions back to the BS.

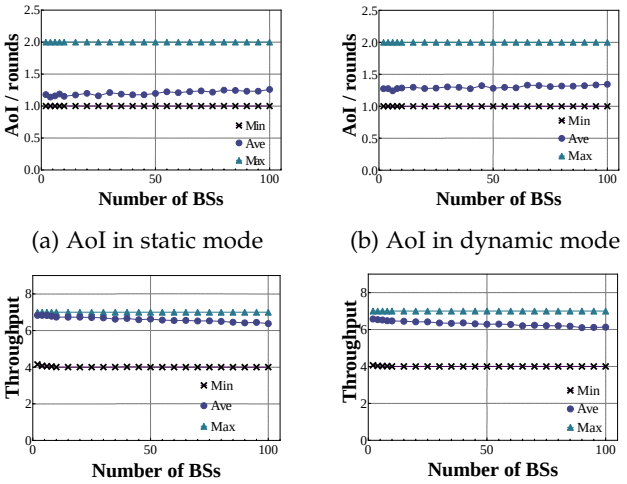
In the following, we further introduce the learning agents implemented in each base station. For T-agents and F-agents, we have two neural networks composed for each of them. Each neural network consists of two fully connected hidden layers. And rectified linear unit is used as the activation function for hidden layers. A memory with 3000 kb is set for each agent. For the learning part, a quadruple (current state, action, reward, next state) is put into the memory after the program is executed by one round. If the size of the memory is exceeded, the earliest quadruple is discarded. In the learning process, we use the mini-batch learning method, and set the size of the batch to 32. When the number of quadruples in memory exceeds the batch size, each training session randomly selects a batch size of quadruples for the training of the agent. In the action decision part, the agent gets an estimate of the value of each action based on the current state through the neural network. Then the ε-greedy strategy is used to decide whether to return the action with the highest estimate or to return a random action. The B-agent maintains a Q-table that stores the expected reward (i.e., Q-value) of each action. When the T-agent and the F-agent output their actions, the B-agent selects the action with the highest current Q-value or executes a random action according to the ε-greedy algorithm. Then it updates the Q-table according to the feedback from the BS and the Equation 13. γ and α in Equation 13 are assigned with the value of 0.95 and 0.1, respectively.

Without loss of generality, over 50 runs of the simulation have been carried out for each reported result. All experiments are conducted on a Linux machine with Intel Xeon CPU E5-2670@2.60GHz, 128 GB main memory, and GPU Nividia RTX 4090. The experiment is implemented in python3 and compiled by a Python compiler.

5.2 Numerical Results

Our simulation results consist of three parts. In part one, we show the performance of our own Deep RL schemes on AoI of all users and throughput of all BSs in static and dynamic modes in Figure 6 and Table 2. In part two, to further investigate the execution of T-, F-, and B-agents, we depict the loss functions of T- and F-agents by Figure 8 , and show the weight parameter of B-agent in Figure 9. Finally, in part three, we compare our solution with two baselines [12, 13], as well as an optimal solution found by brute-force, and the theoretical optimal bound in OMA (Orthogonal Multiple Access) in static and dynamic modes in Figure 10 and 11. In particular, we consider

- Algo1 [12]: each user uses a reinforcement learning approach to choose the most approximate base station from its nearby base stations, to upload its messages by a two-level NOMA technology.



(a) AoI in static mode (b) AoI in dynamic mode

(c) Thr. in static mode (d) Thr. in dynamic mode

Fig. 6: AoI and throughput in the static and dynamic modes

- Algo2 [13]: each user uses a deep reinforcement-learning approach to independently choose the most approximate transmission power from a given power pool, to upload messages to its base station.
- BF: we use a brute force approach to enumerate all the possible actions of BSs in 10 rounds, and choose the one with the minimum average AoI as the optimal action. Then, the optimal action in 10 rounds will be executed repeatedly in comparison. We do not directly choose the optimal actions in a large interval since the running time of our brute force exponentially increases when the number of rounds gets larger.
- OMA: Different from NOMA technology, each base station can receive at most one message in each round with OMA. Thus, there is a natural optimal bound in which each BS only lets the user with the maximal AoI transmit. Such an assumption gives the theoretical upper bound for the throughput and lower bound for AoI minimization in OMA case.

Performance of Our Deep RL Algorithm. Figure 6 and Table 2 show the performance of our own Deep RL scheme on AoI w.r.t. all users and throughput of all BSs in the static and dynamic modes when our learning agents are trained with 4000 rounds. Specifically, in Figure 6(a)-6(b), the x- and y-axes represent the number of BSs and the AoI information, respectively. The minimum/average/maximum AoI in the static and dynamic modes are observed as the number of BSs increasing from 2 to 100. With the same experimental settings, the minimum/average/maximum throughput of BSs are illustrated in Figure 6(c)-6(d), and a detailed data is given in Table 2. According to our observations on the AoI of users and the throughput of BSs, the following results on AoI and throughput can be obtained:

- With respect to the AoI in Figure 6(a)-6(b), when the number of BSs increases from 2 to 100 in both the static and the dynamic mode, the minimum/maximum AoI of users are always 1 and 2 round, respectively. Besides, the average AoI slightly increases but is always smaller than 1.4 round when

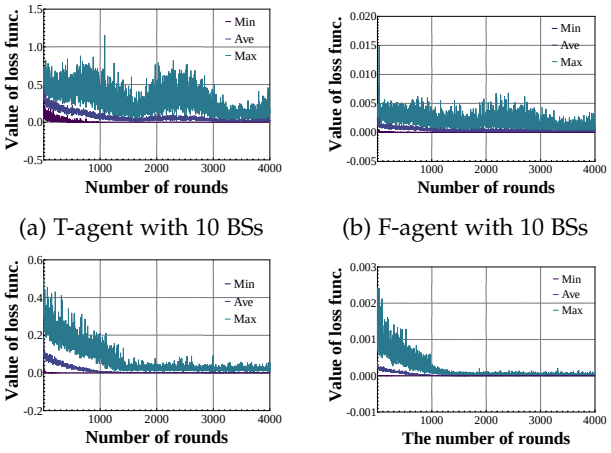
Ratio(%) TP / BS		≤ 3	4	5	6	7	≥ 8
Static mode	[2, 10)	0	0.00	1.31	27.97	70.72	0
	[10, 20)	0	0.01	1.66	30.48	67.85	0
	[20, 30)	0	0.01	1.93	33.09	64.97	0
	[30, 40)	0	0.01	2.57	36.84	60.58	0
	[40, 50)	0	0.02	3.03	38.44	58.51	0
	[50, 60)	0	0.03	4.01	42.54	53.42	0
	[60, 70)	0	0.04	4.37	44.12	51.47	0
	[70, 80)	0	0.05	4.75	45.20	50.00	0
	[80, 90)	0	0.07	6.24	49.09	44.60	0
	[90, 100]	0	0.09	7.08	51.02	41.81	0
Dynamic mode	[2, 10)	0	0.03	4.44	44.59	50.94	0
	[10, 20)	0	0.06	5.32	46.60	48.02	0
	[20, 30)	0	0.07	6.38	49.08	44.47	0
	[30, 40)	0	0.08	7.66	51.82	40.44	0
	[40, 50)	0	0.08	7.37	51.61	40.94	0
	[50, 60)	0	0.13	9.13	54.63	36.11	0
	[60, 70)	0	0.14	9.73	54.75	35.38	0
	[70, 80)	0	0.25	10.97	56.18	32.60	0
	[80, 90)	0	0.39	15.40	58.82	25.39	0
	[90, 100]	0	0.35	14.59	58.88	26.18	0

TABLE 2: Distribution of BSs in terms of the throughput in the static and dynamic modes

there are more BSs implemented in the static and dynamic modes. Additionally, by fixing the number of BSs, the average AoI in the dynamic mode is about 1.1 times larger than that in the static mode. According to our definition on AoI, the inherent lower bound is 1. From the above observations, it can be seen that (1) our proposed algorithm has a high efficiency on minimizing AoI and (2) the performance of our algorithm is not sensitive to the size of the network and mobility of users.

- As for the throughout of BSs illustrated in Figure 6(c)-6(d) and Table 2, its minimum and maximum values are kept at 4 and 7 in most of the times in both of the static and the dynamic modes. Even though the average throughput slightly decreases when the network scope gets larger, due to a heavier global interference, its value is always larger than 6. Note that in OMA technique, the base station can at most receive 1 message in each round, and at most receive 2 messages in the previous works adopting NOMA. The curves on average throughput show that by well-tuning the transmission power levels, each of the BSs in our algorithm can at least receive 6 messages from its users in one round. In other words, the T-agent in our deep DL algorithm works well on improving the throughput of the BSs. Besides, by comparing the minimum/average/maximum throughput in Figure 6, we can see that the gaps between those curves are not large. Thus, it is believed that fairness between BSs has been obtained in our simulation with the help of F-agent. Table 2 gives the distribution of BSs in terms of the throughput in the static and dynamic modes, which further verify the fact that the efficiency and fairness in our AoI scheduling are obtained by T-agent and F-agent, respectively.

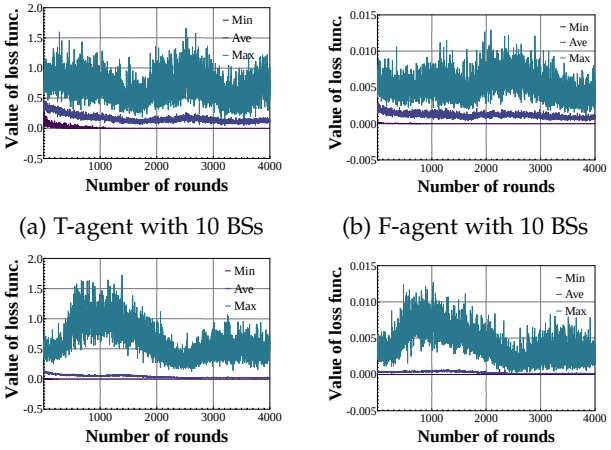
Investigation on Training Process. To further prove that our learning agents finally reach some stable states, the loss



(a) T-agent with 10 BSs (b) F-agent with 10 BSs

(c) T-agent with 100 BSs (d) F-agent with 100 BSs

Fig. 7: Loss functions of T-/F-agents in the static mode



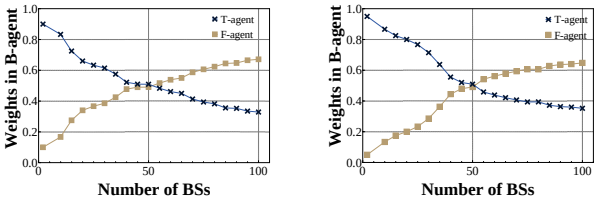
(a) T-agent with 10 BSs (b) F-agent with 10 BSs

(c) T-agent with 100 BSs (d) F-agent with 100 BSs

Fig. 8: Loss functions of T-/F-agents in the dynamic mode

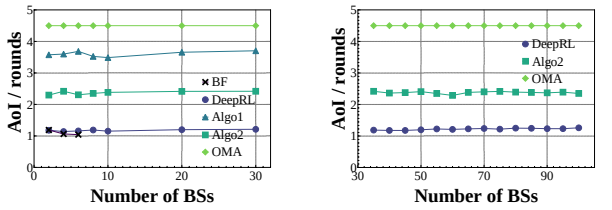
functions of T- and F-agents are illustrated in Figure 7 and Figure 8, in which there are 10 or 100 BSs deployed in the static and dynamic modes. In detail, the x- and y-axes in Figure 7 and Figure 8 represent the number of rounds and the value of loss function, respectively. In each of the subfigures, the minimum, average, and maximum values of loss functions for T-agent and F-agent are observed. From all the curves in Figure 7 and Figure 8, we can see that

- in the static mode when there are 10 or 100 BSs deployed, with the execution of T- and F-agents,

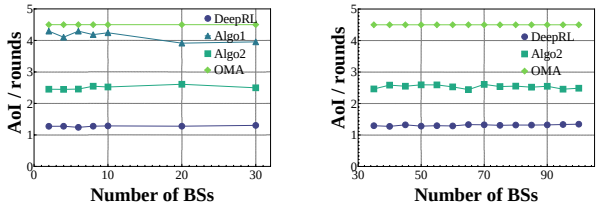


(a) Static mode (b) Dynamic mode

Fig. 9: The weights in B-agent in the static and dynamic modes

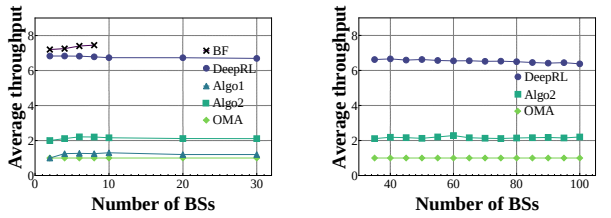


(a) Com. I in static mode (b) Com. II in static mode

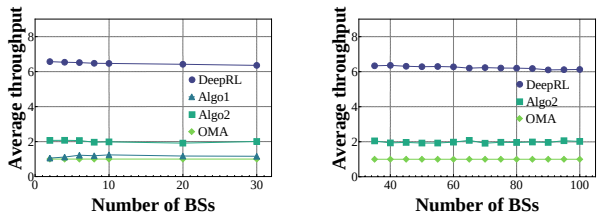


(c) Com. I in dynamic mode (d) Com. II in dynamic mode

Fig. 10: Comparisons on average AoI in the static and dynamic modes



(a) Com. I in static mode (b) Com. II in static mode



(c) Com. I in dynamic mode (d) Com. II in dynamic mode

Fig. 11: Comparisons on average throughput in the static and dynamic modes

the average values of their loss functions gradually reduce and approach to 0 in the training process. The maximum values of these loss functions keep stable at a low level while the minimum values get very close to 0.

- in the dynamic mode with the same deployment of BSs, the average and maximum values of the loss functions from T- and F-agents are also stable, but are relative larger than those in the static mode, which means the mobility of users has the limited impact on the convergence of our algorithm.

From the above results, we can have the conclusion that our T-/F-agents keep stable in their learning process with the mobility of users and various network sizes.

Also, we investigate the weight of T-/F- agents in B-agent when the number of BSs varies in Figure 9, in which the x-axes and y-axes represent the number of BSs and the weight of T-/F-agents in B-agent. From Figure 9, we can see that the weight of T-agent decreases and that of F-agent increases respectively when the number of BSs gets larger in both the static and the dynamic modes. Note that the T-agent on a BS prefers to choose larger power levels for its users, to make sure it has more transmissions succeeded and its local average AoI reduced. The F-agent will choose larger power levels if its local average AoI is larger than that of neighboring BSs, and vice versa. The B-agent balances the actions from T-agent and F-agent to make sure the AoI of all users is as small and also as fair as possible. The curves in Figure 9(a) and 9(b) indicate that when the number of BSs in our simulation is small, the action from T-agent has a heavier weight in B-agent because the interference between users from different BSs is small. Also, when the number of BSs is large, the action from F-agent has a heavier weight in B-agent because too many uplinks ended at different BSs interfere with each other, and the action from F-agent is more important to control and balance the heavy interference.

Comparisons on AoI and Throughput. In Figure 10 and 11, we compare the performance of our scheme with two previous works [12, 13], the optimal solution with NOMA by brute forcing, and the theoretical upper/lower bounds with OMA in the static and the dynamic modes. The algorithms in [12, 13], brute force method, OMA case and our algorithm are termed as Algo1, Algo2, BF, OMA, and DeepRL for short in comparison. Because the running time of BF and Algo1 exponentially increase when the number of BSs and users increases, we test the performance of BF, Algo1, and Algo2 with the number of BSs in the range of $[2, 10]$, $[2, 30]$ and $[2, 100]$, respectively.

In Figure 10(a), we compare the average AoI of all users in our algorithm with that in BF, OMA, Algo1 and Algo2 when the number of BSs varies from 2 to 30. Firstly, the average AoI of uses in our algorithm is close to that in optimal solution by brute force method, which shows the efficiency of our algorithm. Secondly, the average AoI in our algorithm is about 2, 3, and 4 times smaller than those in Algo1, Algo2, and OMA, respectively. This is because the NOMA technology in Algo1 only supports 2-messages synchronously decoded by the BS in each time, Algo2 is considered in a single hop network with one BS without considering the interference from neighboring BSs, and OMA cannot support multiple messages decoded synchronously. While our algorithm is considered for multi-hop networks with numerous BSs and by the well-tuned transmission power levels from deep reinforcement learning, there are 6-7 messages successfully decoded by each BS in one round. In Figure 10(b), we also compare the performance of our algorithm with that in Algo2 and OMA when the number of BSs varies from 30 to 100, which shows that our algorithm has a better performance than Algo2 and OMA in the large scale networks. Figure 10(c) and Figure 10(d) show similar comparisons with Algo1, Algo2 and OMA in the dynamic mode. The curves in Figure 10(c) and Figure 10(d) show that our scheme has a better performance than Algo1, Algo2 and OMA in the dynamic mode when the number of BSs vary from 30 to 100. Note that it is nearly impossible to find the optimal solution by brute force method when users randomly move in each round, the comparison with BF is not considered in the dynamic mode.

Figure 11 shows the comparison results between our work with BF, OMA, Algo1, and Algo2 in terms of throughput when the number of BSs varies from 2 to 100. From Figure 11, we can see that the throughput of our scheme is close to the optimal solution in BF and at least 6/4/3 times faster than that of OMA, Algo1, and Algo2, in both the static and the dynamic mode.

Summary of Simulation. In general, in the first part of our numerical results, Figure 6 and Table 2 show that our algorithm performs stably and achieve good performance on minimizing the average AoI by reaching a balance on optimizing the efficiency and fairness in the uplink scheduling process. In the second part, we extract the key parameters of the T-/F-/B- agents in Figure 7-9, to verify the convergence of our training process and strengthen the explanation of our learning schemes. Finally, we compare our algorithm with some optimal results and previous works in Figure 10 and Figure 11, which shows that the performance of our algorithm is close to the optimal result, and outperform to some previous works, especially in some large-scale edge computing networks.

6 Conclusion and Future Work

We consider distributed optimization of multi-user long-term average Age-of-Information objectives in edge computing networks with NOMA transmission. To solve this challenging non-convex online optimization problem, a distributed deep reinforcement learning-based framework that adopts a novel hierarchical decomposition of decision making is proposed in this paper. We design three different types of distributed agents to learn with respect to the efficiency (T-agent) and fairness (F-agent) of Age-of-Information scheduling, as well as a high-level policy balancing these potentially conflicting design objectives (B-agent). The decomposition in our framework not only leads to improved learning performance, but also provides interpretability. The effectiveness of our solution is demonstrated through extensive evaluations on an edge network simulator with 100 edge devices and 800 end devices. It is shown that our algorithm outperforms previous AoI scheduling with NOMA by $200\%-300\%$ and the optimal solution without NOMA by 400%, and indeed comes very close to an optimal solution with NOMA obtained from brute-force. For future work, we plan to consider other multi-agent RL algorithms, as well as explainable RL algorithms, to solve the AoI minimization problem in edge computing.

Acknowledgment

This work was supported in part by the Federal Ministry of Education and Research (BMBF, Germany) within the 6G Research and Innovation Cluster 6G-RIC under Grant 16KISK020K, the National Natural Science Foundation of China (NSFC) under Grant 62102232, 62122042, 61971269, Shandong Science Fund for Excellent Young Scholars (No.2023HWYQ-007) and Natural Science Foundation of Shandong province under Grant ZR2021QF064.

References

[1] J. Chen, Y. Liu, Q. Chen, X. Lan, G. Cheng, Y. Fu, and Z. Zhang. On the Adaptive AoI-aware Buffer-aided Transmission Scheme for NOMA Networks, in *WCNC*, 2021.

[2] S. Mounchili and S. Hamouda. Better User Clustering Scheme in Distributed NOMA Systems, in *AINA*, 2020.

[3] M. Qu, J. Liu, Jun-Bae Seo, and H. Jin. Distributed Fair Channel Access in NOMA Random Access Systems, in *GLOBECOM*, 2019.

[4] Q. He, D. Yuan and A. Ephremides. Optimizing freshness of information: On minimum age link scheduling in wireless systems, in *WiOpt*, 2016.

[5] I. Kadota, A. Sinha, and E. H. Modiano. Scheduling Algorithms for Optimizing Age of Information in Wireless Networks With Throughput Constraints, in *IEEE/ACM Trans. Netw*, 27(4): 1359–1372, 2019.

[6] I. Kadota, E. Uysal-Biyikoglu, R. Singh, and E. H. Modiano. Minimizing the Age of Information in broadcast wireless networks, in *Allerton*, 2016.

[7] C. Li, S. Li, Y. Chen, Y. Thomas Hou, and W. Lou. AoI Scheduling with Maximum Thresholds, in *INFOCOM*, 2020.

[8] Cho-Hsin Tsai and Chih-Chun Wang. Unifying AoI Minimization and Remote Estimation - Optimal Sensor/Controller Coordination With Random Two-Way Delay, in *IEEE/ACM Trans. Netw*, 30(1): 229–242, 2022.

[9] V. Tripathi and E. H. Modiano. An Online Learning Approach to Optimizing Time-Varying Costs of AoI, in *MobiHoc*, 2021.

[10] F. Dressler et al. V-Edge: Virtual Edge Computing as an Enabler for Novel Microservices and Cooperative Computing, in *IEEE Network*, 36(3): 24-31, 2022.

[11] Y. Xie, L. Shi, Z. Wei, J. Xu, Y. Zhang, An energy-efficient resource allocation strategy in massive MIMO-enabled vehicular edge computing networks, in *High-Confidence Computing*, 3: (3), 2023.

[12] W. Ahsan, W. Yi, Z. Qin, Y. Liu, and A. Nallanathan. Resource Allocation in Uplink NOMA-IoT Networks: A Reinforcement-Learning Approach, in *IEEE Trans. Wirel. Commun*, 20(8): 5083–5098, 2021.

[13] M. Fayaz, W. Yi, Y. Liu, and A. Nallanathan. Transmit Power Pool Design for Grant-Free NOMA-IoT Networks via Deep Reinforcement Learning, in*IEEE Trans. Wirel. Commun*, 20(11): 7626–7641, 2021.

[14] B. A. G. R. Sharan, S. Deshmukh, S. R. B. Pillai, B. Beferull-Lozano. Energy Efficient AoI Minimization in Opportunistic NOMA/OMA Broadcast Wireless Networks, in *IEEE Trans. Green Commun. Netw.*, 6(2): 1009-1022, 2022.

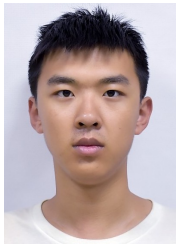
[15] H. B. Beytur and E. Uysal-Biyikoglu. Age Minimization of Multiple Flows using Reinforcement Learning, in *ICNC*, 2019.

[16] E. T. Ceran, D. Gündüz, and A György. Average Age of Information With Hybrid ARQ Under a Resource Constraint, in *IEEE Trans. Wirel. Commun*. 18(3): 1900–1913, 2019.

[17] E. T. Ceran, Deniz Gündüz, and A. György. Reinforcement Learning to Minimize Age of Information with an Energy Harvesting Sensor with HARQ and Sensing Cost, in *INFOCOM*, 2019.

[18] E. T. Ceran, D. Gündüz, and A. György. A Reinforcement Learning Approach to Age of Information in Multi-User Networks With HARQ, in *IEEE J. Sel. Areas Commun*,

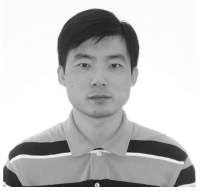
39(5): 1412–1426, 2021.
[19] G. Zhang, C. Shen, Q. Shi, B. Ai, Z. Zhong. AoI Minimization for WSN Data Collection With Periodic Updating Scheme, in *IEEE Trans. Wirel. Commun.*, 22(1): 32-46, 2023.
[20] H. Lv, Z. Zheng, F. Wu, and G. Chen. Strategy-Proof Online Mechanisms for Weighted AoI Minimization in Edge Computing, in *IEEE J. Sel. Areas Commun*, 39(5): 1277–1292, 2021.
[21] J. Zhu and J. Gong. Online Scheduling of Transmission and Processing for AoI Minimization with Edge Computing, in *CoRR* abs/2202.06193, 2022.
[22] Yu-Pin Hsu, E. H. Modiano, and L. Duan. Age of information: Design and analysis of optimal scheduling algorithms, in *ISIT*, 2017.
[23] I. Kadota, A. Sinha, E. Uysal-Biyikoglu, R. Singh, and E. H. Modiano. Scheduling Policies for Minimizing Age of Information in Broadcast Wireless Networks, in *IEEE/ACM Trans. Netw*, 26(6): 2637–2650, 2018.
[24] S. Leng and A. Yener. Age of Information Minimization for Wireless Ad Hoc Networks: A Deep Reinforcement Learning Approach, in *GLOBECOM*, 2019.
[25] E. Altman, R. E. Azouzi, D. S. Menasché, and Yuedong Xu. Forever Young: Aging Control In DTNs, in *CoRR* abs/1009.4733, 2010.
[26] S. K. Kaul, R. D. Yates, and M. Gruteser. Real-time status: How often should one update? in *INFOCOM*, 2012.
[27] R. D. Yates, Y. Sun, D. R. Brown III, S. K. Kaul, E. H. Modiano, S. Ulukus. Age of Information: An Introduction and Survey, in *CoRR* abs/2007.08564 (2020).
[28] J. Lou, X. Yuan, S. Kompella, and Nian-Feng Tzeng. AoI and Throughput Tradeoffs in Routing-aware Multi-hop Wireless Networks, in *INFOCOM*, 2020.
[29] R. Talak and S. Karaman and E. H. Modiano. Distributed Scheduling Algorithms for Optimizing Information Freshness in Wireless Networks, in *SPAWC*, 2018.
[30] R. Talak, S. Karaman, and E. H. Modiano. Minimizing age-of-information in multi-hop wireless networks, in *Allerton*, 2017.
[31] R. Talak, S. Karaman, and E. H. Modiano. Optimizing Information Freshness in Wireless Networks Under General Interference Constraints, in *IEEE/ACM Trans. Netw*, 28(1): 15–28, 2020.
[32] Q. Liu, H. Zeng, M. Chen. Minimizing AoI With Throughput Requirements in Multi-Path Network Communication, in *IEEE/ACM Trans. Netw.*, 30(3): 1203-1216, 2022.
[33] C. Luo, Y. Hou, Y. Hong, Z. Chen, N. Liu, D. Li. AoI Minimizing of Wireless Rechargeable Sensor Network Based on Trajectory Optimization of Laser-Charged UAV, in *AAIM*, 255-267, 2022.
[34] P. Zou, O. Ozel, S. Subramaniam. Optimizing Information Freshness Through Computation-Transmission Tradeoff and Queue Management in Edge Computing, in *IEEE/ACM Trans. Netw.*, 29(2): 949-963, 2021.
[35] L. Liu, K. Xiong, J. Cao, Y. Lu, P. Fan, K. B. Letaief. Average AoI Minimization in UAV-Assisted Data Collection With RF Wireless Power Transfer: A Deep Reinforcement Learning Scheme, in *IEEE Internet Things J.*, 9(7): 5216-5228, 2022.
[36] A. H. Zarif, P. Azmi, N. M. Yamchi, M. R. Javan, E. A. Jorswieck. AoI Minimization in Energy Harvesting and Spectrum Sharing Enabled 6G Networks, in *IEEE Trans. Green Commun. Netw.*, 6(4): 2043-2054, 2022.
[37] S. Feng, J. Yang. Optimal status updating for an energy harvesting sensor with a noisy channel, *INFOCOM Workshops*, 2018.
[38] Y. Sun, E. Uysal-Biyikoglu, R. D. Yates, C. E. Koksal, N. B. Shroff. Update or Wait: How to Keep Your Data Fresh, in *IEEE Trans. Inf. Theory*, 63(11): 7492-7508, 2017.
[39] T. Zhu, J. Li, H. Gao, Y. Li, Z. Cai. AoI Minimization Data Collection Scheduling for Battery-Free Wireless Sensor Networks, in *IEEE Transactions on Mobile Computing*, early access article, DOI: 10.1109/TMC.2021.3106013, 2021.
[40] F. Zhao, X. Sun, W. Zhan, X. Wang, X. Chen. Information Freshness in Random-Access Poisson Network: Average AoI versus Peak AoI, in *VTC Fall*, 1-6, 2022.
[41] T. Haarnoja, A. Zhou, P. Abbeel and S. Levine. Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor, in *CoRR* abs1801.01290, 2018.
[42] P. Christodoulou, Soft Actor-Critic for Discrete Action Settings, in *CoRR* abs/1910.07207, 2019.



Congwei Zhang received the B.E. degree from the Computer Science Program of Taishan College, Shandong University, in 2021. He is currently pursuing the Ph.D. degree with the School of Computer Science and Technology, Shandong University. His research interests include wireless networks, edge intelligence, and distributed computing.distributed computing.

Yifei Zou received the B.E. degree in 2016 from Computer School, Wuhan University, and the PhD degree in 2020 from the Department of Computer Science, The University of Hong Kong. He is currently an Assistant Professor with the school of computer science and technology, Shandong University. His research interests include wireless networks, ad hoc networks and distributed computing.

Zuyuan Zhang received the B.S. degree from the Shandong University, China in 2023. He is currently a second year Ph.D. student and a Research Assistant in Electrical and Computer Engineering deparment at The George Washington University. His research interests include Optimization and Game Theory.

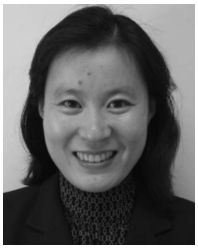


Dongxiao Yu received the BSc degree in 2006 from the School of Mathematics, Shandong University and the PhD degree in 2014 from the Department of Computer Science, The University of Hong Kong. He became an associate professor in the School of Computer Science and Technology, Huazhong University of Science and Technology, in 2016. He is currently a professor in the School of Computer Science and Technology, Shandong University. His research interests include wireless networks, distributed computing and graph algorithms.

Jorge Torres Gómez is with the School of Electrical Engineering and Computer Science, TU Berlin. From 2008 to 2018, he lectured at the School of Telecommunications and Electronics, Technological University José Antonio Echeverría in Cuba. He has been with the Department of Signal Theory and Communications, UC3M, Spain, as a guest lecturer and with the TU Chemnitz as a postdoc. He is on the Educational and Professional Activities committee in the IEEE German Chapter Executive Committee. His research interests include wireless communications, molecular communications, DSP, SDR, and Education.

Tian Lan received the B.A.Sc. degree from the Tsinghua University, China in 2003, the M.A.Sc. degree from the University of Toronto, Canada, in 2005, and the Ph.D. degree from the Princeton University in 2010. Dr. Lan is currently a full Professor of Electrical and Computer Engineering at the George Washington University. His research interests include network optimization, algorithms, and machine learning. He received the Meta Research Award in 2021, SecureComm Best Paper Award in 2019, SEAS Faculty Recognition Award in 2018, Hegarty Faculty Innovation Award in 2017, AT&T VURI Award in 2015, IEEE INFOCOM Best Paper Award in 2012, Wu Prizes for Excellence at Princeton University in 2010, IEEE GLOBECOM Best Paper Award in 2009, and IEEE Signal Processing Society Best Paper Award in 2008.

Falko Dressler received his M.Sc. and Ph.D. degrees from the Dept. of Computer Science, University of Erlangen in 1998 and 2003, respectively. He is a full professor and Chair for Data Communications and Networking at the School of Electrical Engineering and Computer Science, TU Berlin. Dr. Dressler has been associate editor-in-chief for IEEE Trans. on Mobile Computing and Elsevier Computer Communications as well as an editor for journals such as IEEE/ACM Trans. on Networking, IEEE Trans. on Network Science and Engineering, Elsevier Ad Hoc Networks, and Elsevier Nano Communication Networks. He has been chairing conferences such as IEEE INFOCOM, ACM MobiSys, ACM MobiHoc, IEEE VNC, IEEE GLOBECOM. He authored the textbooks Self-Organization in Sensor and Actor Networks published by Wiley & Sons and Vehicular Networking published by Cambridge University Press. He has been an IEEE Distinguished Lecturer as well as an ACM Distinguished Speaker. Dr. Dressler is an IEEE Fellow as well as an ACM Distinguished Member. He is a member of the German National Academy of Science and Engineering (acatech). He has been serving on the IEEE COMSOC Conference Council and the ACM SIGMOBILE Executive Committee. His research objectives include adaptive wireless networking (radio, visible light, molecular communications) and embedded system design (from microcontroller to Linux kernel) with applications in ad hoc and sensor networks, the Internet of Things, and cooperative autonomous driving systems.



Xiuzhen Cheng received her M.S. and Ph.D. degrees in computer science from the University of Minnesota – Twin Cities in 2000 and 2002, respectively. She is a professor in the School of Computer Science and Technology, Shandong University. Her current research interests include cyber physical systems, wireless and mobile computing, sensor networking, wireless and mobile security, and algorithm design and analysis. She has served on the editorial boards of several technical journals and the technical program committees of various professional conferences/workshops. She also has chaired several international conferences. She worked as a program director for the US National Science Foundation (NSF) from April to October in 2006 (full time), and from April 2008 to May 2010 (part time). She received the NSF CAREER Award in 2004. She is Fellow of IEEE and a member of ACM.

APPENDIX

(1) The RL technique is powerful to solve the network optimization problem. However, the MLA-AoI cannot be directly used as the final objective of the RL. In the revised version, we have explained the motivation of adopting the distributed RL technique to optimize the MLA-AoI problem in mobile network. Whereas, the MLA-AoI cannot be directly used as the final objective of the RL, for the following two reasons. **Reason 1:** as defined in the equation (2) of model section, the MLA-AoI is the average AoI of multi-users in an interval I, which is a time accumulative metric. Since the states, rewards, and actions in RL are mainly roundly based, setting the MLA-AoI as the final reward may reduce the accuracy of learning. For example, when the MLA-AoI in the previous rounds are very large, even though the RL technique chooses an optimal action in current round, the new MLA-AoI may decrease a little. In this case if the MLA-AoI is set as the reward of RL, the learning process will be misled since the optimal action only gains a little reward. Thus, a new metric that is roundly-independent should be used as the reward of RL to evaluate the actions in each round. **Reason 2:** the MLA-AoI is the averaged long term AoI of multi-users, which is a global metric. If the MLA-AoI is adopted as the final reward of RL, it is time and resource consuming to aggregate the average long term AoI from all the users in a multihop wireless network. In other words, the efficiency of RL will be heavily delayed. Because of these two reasons, the MLA-AoI is not an appropriate option as the reward of RL.

(2) In a single hop wireless network, maximizing the throughput of BS in each round is equivalent to minimizing the MLA-AoI. We consider a single hop wireless network that consises of 1 BS and n users. As defined in the model section, $\mathcal{A}_v(t)$ is the AoI of the user v at round t with $\mathcal{A}_v(0) = 1$ initially. We consider an action that guarantees k communications succeeded with power levels $\{k, k-1, ..., 1\}$ in each round.[3] Let V be the set of n users, and $V_k(t)$ be the set of k users that have the largest AoI at the beginning of round t. By letting the k users in set $V_k(t)$ transmit with power levels $\{k, k-1, ..., 1\}$ respectively and other nodes listen, we know that the AoI of uses in set $V_k(t)$ reduces to 1 and the AoI of uses outside the set $V_k(t)$ increases by 1. Let

3. For a brief analyze, we assume that n/k is an integer. Otherwise, the integers $\lfloor n/k \rfloor$ and $\lceil n/k \rceil$ will be used in analyze.

$\tilde{A}(t)$ be the averaged AoI of multi-users at round t. We have $\tilde{A}(t) = 1$ when $t = 0$, and

$$\begin{aligned}\tilde{A}(t) &= \frac{\sum_{v\in V} A_v(t)}{n} = \frac{\sum_{v\in V\setminus V_k(t)} A_v(t) + \sum_{v\in V_k(t)} 1}{n} \\ &= \frac{\sum_{v\in V} \left(A_v(t-1)+1\right) - \sum_{v\in V_k(t)} A_v(t-1)}{n} \\ &= \tilde{A}(t-1) + 1 - \frac{\sum_{v\in V_k(t)} A_v(t-1)}{n} \quad \text{when} \quad t > 0\end{aligned}$$

Additionally, for each node $v \in V_k(t)$, we have

$$A_v(t) = \begin{cases} t+1 & 0 \le t \le n/k \\ n/k & t > n/k \end{cases}$$

Thus, when $0 \le t \le n/k$ we further have

$$\begin{aligned}\tilde{A}(t) &= \tilde{A}(t-1) + (1 - \frac{tk}{n}) \\ &= \tilde{A}(t-2) + (1 - \frac{(t-1)k}{n}) + (1 - \frac{tk}{n}) \\ &= ... \\ &= A(0) + (1 - \frac{k}{n}) + (1 - \frac{2k}{n}) + ... + (1 - \frac{tk}{n}) \\ &= A(0) + \sum_{i=1}^{t}(1 - \frac{ik}{n}) = 1 + t - \frac{kt(1+t)}{2n}\end{aligned}$$

and when $t > n/k$,

$$\tilde{A}(t) = \tilde{A}(t-1) + 1 - \frac{\sum_{v\in V_k(t)} A_v(t-1)}{n} = \tilde{A}(t-1)$$

According to the equation (2) in model section, for an interval I starting from round 0 and ending at round x_0, we have when $0 \le x_0 \le n/k$

$$\tilde{A}(I) = \frac{1}{x_0}\sum_{t=0}^{x_0}\left(1 + t - \frac{kt(1+t)}{2n}\right) \tag{14}$$

, and when $x_0 > n/k$

$$\tilde{A}(I) = \frac{1}{x_0}\left(\sum_{t=0}^{n/k}\left(1 + t - \frac{kt(1+t)}{2n}\right) + \sum_{t=n/k}^{x_0}\left(\frac{1}{2} + \frac{n}{2k}\right)\right). \tag{15}$$

From the above equation, we can see that when the throughput k is larger, the MLA-AoI $\tilde{A}(I)$ is smaller.

(3) In a multi-hop wireless network, each BS maximizing its throughput may result in a Prisoner Dilemma, which can be solved by considering the fairness. The multi-hop scenario can be divided into multiple single-hop wireless networks. As what we have proved in the single hop scenario, each BS can maximize its throughput to reduce its MLA-AoI. Whereas, for a BS A, no matter which power levels its neighboring BSs adopted, adopting the largest power levels is always A's (local) optimal solution. For example, we consider two BS A and B with parameters $d(A,B) = 2$, $\alpha = 3$, $N = 1$ five layers power level with $P_i = 2.1^{i-1} \times d(A,B)^{\alpha} \times N$ for $i = 0, 1, 2, 3, 4$. Each BS has 5 users. The action of BS A is denoted by a variable x. $x = 2$ means the power levels $\{1, 2\}$ will be used by the BS A. Similarly, y denotes the action of BS B. For any given x and y, we can figure out the throughput (k_a, k_b) of BS A and B, as is listed in the Table 3. From the Table 3, we can see that no matter which action BS B taken, $x = 5$ is always the local optimal action for BS A. Similar result can be concluded for the BS B with the optimal action $y = 5$. Whereas, when both of BS A and B take 5 levels transmission power, their local throughput and the global throughput are not the optimal. [4] To avoid such a Prisoner Dilemma, the fairness issue is considered in our RL algorithm design, which requires that the average AoI of BS A and B should not differ too much. Such a fairness scheme helps the BS A and B to move out from the action pair $x = 5$ and $y = 5$. Specifically, when BS A and B choose the action pair $x = 5$ and $y = 5$, their throughputs are $k_a = 2$ and $k_b = 2$. When BS A choose $x = 3$ as the exploration in RL, their throughputs are $k_a = 0$ and $k_b = 5$. Consider the throughput and fairness simultaneously, the BS B will also choose $y = 3$, which is a new Nash equilibrium in terms of throughput and fairness.

(k_a, k_b) x \ y	1	2	3	4	5
1	(1,1)	(1,2)	(1,3)	(0,4)	(0,5)
2	(2,1)	(2,2)	(2,3)	(0,4)	(0,5)
3	(3,1)	(3,2)	(3,3)	(2,4)	(0,5)
4	(4,0)	(4,0)	(4,2)	(2,2)	(1,4)
5	(5,0)	(5,0)	(5,0)	(4,1)	(2,2)

TABLE 3: Example of throughput of two base stations with different number of NOMA layers

In this paper, optimizing the throughput and fairness simultaneously can be a heuristic solution to minimize the MLA-AoI. In conclusion, MLA-AoI cannot be directly used as the final objective in our distributed RL framework. Maximizing the throughput of BS in each round is equavalent to minimizing the MLA-AoI in single hop wireless network. Whereas, in a multihop wireless network, each BS maximizing its throughput results in a Prisoner Dilemma. Thus, the fairness is considered as one of the objectives, to avoid the Prisoner Dilemma. With the above consideration, we design T-agent on each BS to optimize its throughput, F-agent for the fairness, and B-agent to reach a balance on the actions from T and F-agents. Even though our approach is a heuristic solution to minimize the MLA-AoI, a clear and reasonable algorithm designing is presented. Besides, the numerical results show that the performance of our algorithm is not far away from the optimal solution.

4. Local throughput means the throughpt of BS A or B, the global throughput indicates the sum of the throughputs from BS A and B.