

Distributed Fast and Fair Physical-to-Virtual Communications in Digital Twin Edge Networks

Senmao Qi, Yifei Zou, *Member, IEEE*, Li Yang, Jiguo Yu, *Fellow, IEEE*, Xiuzhen Cheng, *Fellow, IEEE*, Falko Dressler, *Fellow, IEEE*, and Dongxiao Yu, *Senior Member, IEEE*

Abstract—Physical-to-Virtual Communication (PTVC) is essential for Digital Twin Edge Networks (DITENs) to synchronize physical objects with their digital twins. In practice, update messages from physical objects to their digital twins often have different priorities based on their urgency, necessitating priority-aware scheduling to ensure that critical messages are updated with low latency. While existing approaches can achieve fairness by collecting global priority information and solving optimization problems, when the number and priorities of nodes change, it takes substantial communication resources to timely update the global priority information in large-scale edge networks. To address these limitations, we propose DFF-PTVC, a fully distributed fast and fair communication algorithm where each end node makes scheduling decisions without requiring knowledge of other nodes' priorities. Through an edge-end collaborative framework, the edge server adaptively broadcasts a meta probability to manage channel contention, while each end node autonomously determines its transmission probability by combining the meta probability with its local priority. We rigorously prove that DFF-PTVC achieves asymptotically optimal throughput of $\Omega(1)$, guarantees expected latency of $O(Kn/k)$ for messages with priority k , and ensures weighted fairness, where n is the number of end nodes and K is the upper bound of message priority. We further develop DFF-PTVC-DRL, a deep reinforcement learning variant for enhanced performance when the edge server has sufficient computational resources. Extensive simulations demonstrate that our distributed algorithms, using only local priority information, achieve performance within a small constant factor of the centralized optimal solution that requires global network knowledge.

Index Terms—Digital Twin Edge Networks, Efficient Physical-to-Virtual Communication, Fairness, Distributed Algorithms.

I. INTRODUCTION

This work was supported in part by the National Natural Science Foundation of China under Grants U24A20244.

S. Qi is with the State Key Laboratory of Blockchain and Data Security, and the College of Computer Science and Technology, Zhejiang University, Hangzhou, 310007, China. E-mail: senmao_qi@zju.edu.cn.

Y. Zou (Corresponding Author) and X. Cheng are with School of Computer Science and Technology, and the Shandong Provincial Key Laboratory of Computing-Network Integration, Shandong University, Qingdao, 266237, China. E-mail: yfzou@sdu.edu.cn, xzcheng@sdu.edu.cn.

D. Yu is with the School of Cryptologic Science and Engineering, and the State Key Laboratory of Cryptography and Digital Economy Security, Shandong University, Jinan, 250100, China. E-mail: dxyu@sdu.edu.cn.

L. Yang is with the College of Computer Science and Technology, Qingdao University, Qingdao, 266071, P.R. China. E-mail: yangli1@qdu.edu.cn.

J. Yu is with School of Information and Software Engineering, University of Electronic Science and Technology of China, Chengdu, 611731, China, and also with the Big Data Institute, Qilu University of Technology, Jinan, 250353, China. E-mail: jiguo.yu@sina.com.

F. Dressler are with the School of Electrical Engineering and Computer Science, TU Berlin, Berlin, 10587, Germany. E-mail: dressler@ccs-labs.org.

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

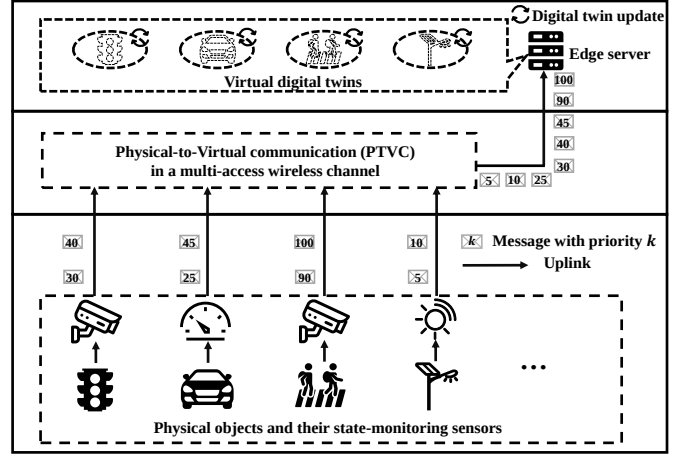


Fig. 1: An illustration of physical-to-virtual communication (PTVC) in digital twin edge networks (DITENs). In uplink communications, messages with different priorities collected by sensors are transmitted from physical objects to update their corresponding digital twins at the edge server.

By implementing digital twins of physical objects on an edge server, digital twin edge networks (DITENs) provide customized services with fast insight and low-latency to users, and have wide applications in Internet-of-Things networks [1], vehicular networks [2], and healthcare systems [3]. Physical-to-virtual communication (PTVC) is an essential part of DITENs [4], through which edge-side digital twins can be timely updated according to the information of physical objects collected by the sensors, which are also termed as end nodes in edge networks. When designing PTVC algorithms, efficiency and fairness are important considerations. As illustrated in Figure 1, the former ensures a low-latency and high-throughput message updating from physical objects to the digital twins [5], [6]. Considering that the updating messages may have different priorities according to their urgency, the latter focuses on a fair scheduling of updating messages according to their priority levels, i.e., higher-priority messages should be received faster by the server [7].

Due to its significance, some communication algorithms have been proposed to achieve these goals in twin updating. These include optimization-based [8]–[11], game theory-based [12]–[14], and machine learning-based [15]–[18] strategies that allocate network resources such as bandwidth and transmission power to improve efficiency. Besides efficiency, some studies further introduce weighted metrics [7], [19]–

[21] to additionally consider fairness in communication. For instance, Feng *et al.* [7] propose a QoE fairness-aware resource allocation strategy for wireless virtual reality systems in DITENs, which jointly optimizes bandwidth allocation and transmission power to balance service latency and fairness among users. Despite the aforementioned studies, several concerns remain unsolved. Firstly, existing fairness-aware strategies [7], [19], [20] are designed under a static priority assumption, which is not flexible enough when nodes' priorities¹ vary over time in practical edge environments. Secondly, most of them require global information about priorities of updating messages as preliminary knowledge for message scheduling [8]–[11], [15]–[17]. However, when the number of nodes grows or priorities change dynamically in large-scale distributed edge networks, real-time collecting and updating global priority information across the network incurs prohibitive communication costs. Furthermore, in distributed edge environments, systems relying on centralized coordination may suffer from single points of failure and limited system adaptability [22].

This paper studies the efficient and fair PTVC problem in digital twin edge networks, in which the priorities of nodes fluctuate when important/urgent information is collected from the environment, or delivered to the edge server. To address these unresolved concerns, we target a fully distributed communication algorithm that operates with purely local information at each node, i.e. the global priority information is not required in message scheduling.

Solving the PTVC problem while meeting the above requirements is not easy, due to the following issues. The **first challenge** is how to balance efficiency and fairness in scheduling when the priorities of end nodes vary with time. This challenge concerns the efficiency-fairness trade-off under dynamic priorities. On the one hand, excessive contention on the wireless channel makes the simultaneous communications physically interfere with each other and fail due to collisions [23]. On the other hand, low contention on the wireless channel may lead to under-utilization and reduced efficiency. Therefore, balancing the contention for communication efficiency is not an easy task in a distributed manner, let alone ensuring fairness for end nodes with time-varying priorities. The **second challenge** is how to achieve priority-aware fairness using only local information. This challenge is about how to realize such fairness when each node only knows its own local priority. To ensure that higher-priority messages are received faster by the server, each node should adjust its transmission behavior according to its relative priority among all competing nodes. Without a global view on the priority information, a node cannot determine whether to compete aggressively or conservatively for the channel, making it difficult to achieve priority-aware scheduling where higher-priority messages obtain lower-latency delivery.

To overcome the challenges mentioned above, a distributed fast and fair PTVC algorithm (DFF-PTVC) is proposed. Specifically, DFF-PTVC follows a round-based communica-

tion structure, where time is divided into equal-length slots for message exchanges between end nodes and the edge server. In each round, a priority-based broadcast strategy is used at the end nodes for fairness, and an adaptive strategy is employed at the edge server for efficiency. These strategies are detailed below. **Ⓐ Broadcast Strategy at the End Side for Fairness.** In each round, the edge server broadcasts a meta probability p_e to all end nodes. Upon receiving the meta probability p_e , each end node v_i with priority w_i transmits its messages with probability $w_i \cdot p_e$. The larger the priority of node v_i , the more important its messages are, and the more likely they are to be scheduled, which guarantees fairness in PTVC. The probability p_e is termed as the meta probability because all end nodes derive their transmission probabilities from it. **Ⓑ Adaptive Strategy at the Edge Side for Efficiency.** When the end nodes transmit, the edge server senses the channel state and adaptively adjusts the meta probability based on the contention level in the wireless channel. As proven in the analysis section, our adaptive strategy guarantees a constant throughput in PTVC, which is asymptotically optimal in terms of efficiency. The collaboration between the end nodes and the edge server in DFF-PTVC adopts broadcast and adaptive strategies, simultaneously optimizing efficiency and fairness in PTVC. Since the broadcast strategy is directly implemented at each end node, its parameters can be updated in a timely manner as node priorities change over time, which enhances the flexibility of our algorithm.

Considering the advantage of artificial intelligence in handling complex and dynamic scenarios, we further develop our adaptive strategy from a rule-based approach to a DRL-based (deep reinforcement learning-based) strategy (DFF-PTVC-DRL) that learns fine-grained control policies directly from network observations, enabling more sophisticated and adaptive tuning of the meta probability in response to time-varying channel conditions and priority dynamics. The contributions are summarized as follows.

- This paper studies the PTVC problem in DITENs with joint consideration of efficiency and fairness. Unlike existing work that relies on global priority information [7], [19]–[21], we tackle the PTVC problem under a more practical yet challenging setting where each node has access only to its local priority information.
- We propose a distributed fast and fair physical-to-virtual communication algorithm (DFF-PTVC), which consists of a broadcast strategy at the end nodes for fairness and an adaptive strategy at the edge server for efficiency. To address the challenge of achieving fairness without global priority knowledge, our algorithm allows each end node to get an appropriate scheduling priority using only its local priority and a coordination signal from the edge server. We rigorously prove that DFF-PTVC guarantees: (1) **efficiency**: expected latency of $O(Kn/k)$ rounds for priority- k messages, maximum weighted latency of $O(Kn)$ rounds, and asymptotically optimal throughput of $\Omega(1)$ per round, where n is the number of end nodes and K is the upper bound of message priority; (2) **weighted fairness**: the probability ratio of any two messages being

¹The priority of an end node is determined by the priorities of its updating messages.

received by the edge server is proportional to their priority ratio. All results hold with high probability².

- By replacing the rule-based adaptive strategy with a deep reinforcement learning (DRL)-based one, we extend our work to develop an AI-enabled distributed algorithm, named DFF-PTVC-DRL, which is more refined and effective in addressing complex edge scenarios involving time-varying priorities.

Necessary comparisons and scalability studies are conducted to validate the performance of our algorithm in terms of multiple key metrics, including average latency, maximum latency, throughput, and Jain's fairness index.

II. RELATED WORK

As a combination of digital twin technology and edge computing, Digital Twin Edge Networks (DITENs) enable real-time monitoring and intelligent decision-making across various domains [1], [3], [31]. In DITENs, accurate and timely updates from physical devices to their digital counterparts—termed Physical-to-Virtual Communications (PTVC)—are critical for maintaining synchronization between the physical and digital worlds [4]. Prior work on PTVC has primarily focused on improving communication efficiency and fairness through various scheduling and resource allocation strategies, which are summarized as follows.

Centralized Approaches. Much of the existing work adopts centralized coordination that collects global network information for scheduling decisions. Optimization-based methods [8]–[11] formulate PTVC as resource allocation problems requiring global state information, but lack flexibility for dynamic environments. To improve adaptability, deep reinforcement learning strategies [15]–[18] learn policies from historical data, yet still rely on centralized training with global observations. Recognizing priority heterogeneity, fairness-aware approaches [7], [19], [20] introduce weighted fairness mechanisms through centralized max-min optimization. Recently, age of information (AoI) has been adopted as a freshness metric for twin updating, and several centralized scheduling schemes have been proposed accordingly, including AoI-aware service provisioning for digital twin network slicing [24], joint AoI-energy optimization for twin synchronization [25], and AoI-aware scheduling for industrial IoT edge networks [26]. These methods improve the timeliness of twin updates, but still rely on a central controller that collects global network state. However, when message priorities change dynamically, maintaining global information incurs significant communication overhead in these centralized methods.

Distributed Approaches. Other work adopts distributed coordination without centralized controllers. Game-theoretic approaches [12]–[14] iteratively exchange neighbor information to reach Nash equilibrium for efficiency optimization, while consensus-based methods [21] employ distributed optimization where nodes repeatedly exchange and update variables with neighbors until reaching agreement on fairness-aware resource allocation. However, these methods still require iterative exchange of priority-related or control information beyond

the local node, which incurs substantial coordination overhead when node priorities change frequently. More generally, classical random-access and AoI-aware scheduling schemes also provide useful references for distributed PTVC design. Slotted ALOHA [27] and exponential back-off mechanisms resolve contention using local random access, but they do not differentiate messages by priority. IEEE 802.11e EDCA [28] introduces priority differentiation through contention-window tuning, yet it remains a per-node MAC mechanism and does not target weighted fairness in our PTVC setting. AoI-aware schemes such as ADRA [29] improve timeliness by gating transmissions on local message age, but remain agnostic to heterogeneous message weights. Distributed AoI optimization methods such as [30] further exploit additional global weight information to improve scheduling, at the cost of extra coordination overhead.

Overall, existing methods either rely on information exchange beyond the local node or remain weight-agnostic in scheduling. In contrast, this paper proposes a distributed communication algorithm where each node makes scheduling decisions using only its local priority and a broadcast meta probability from the edge server, without requiring knowledge of other nodes' priorities. Our design enables rapid adaptation to dynamic priority changes in practical edge environments. A detailed comparison with representative methods is provided in Table I.

III. SYSTEM MODEL AND PROBLEM DEFINITION

We consider a digital twin edge network consisting of n physical devices as end nodes and an edge server hosting their digital twins. The end nodes constantly collect physical information from the environment, generating priority-weighted messages that must be transmitted to the edge server to update their digital twins. A multi-access wireless channel is provided for communication, where simultaneous transmissions may fail due to physical interference. This part first introduces the network architecture and communication model, then describes the message generation process, and finally formulates the weighted message scheduling problem.

System Model for DITEN. We assume that all the end nodes and the edge server in DITEN are deployed in a single hop wireless environment with an arbitrary spatial distribution. Variable v_e , sets $V = \{v_1, v_2, \dots, v_n\}$, and $\bar{V} = \{\bar{v}_1, \bar{v}_2, \dots, \bar{v}_n\}$ are used to denote the edge server, the end nodes, and their corresponding digital twins, respectively. Each end node v_i maintains a local message queue Q_i that stores messages waiting to be transmitted to the edge server. When a message is successfully received by the server and acknowledged, it is removed from the queue. The weight of a node v_i , denoted as w_i , is defined as the sum of the weights of all messages currently in its queue Q_i .

Communication Model. Communication time in our system is divided into successive *rounds*, each of which consists of multiple discrete *time slots*. A time slot represents the minimal duration required for a single message transmission between end nodes and the edge server. A similar simplification can also be found in [4], [23], [32]. By doing this,

²Specifically, with probability at least $1 - n^{-c}$ for some constant $c > 1$.

TABLE I: COMPARISON OF SCHEDULING AND RESOURCE ALLOCATION APPROACHES FOR PTVC IN DITENS

Category	Representative Work	Efficiency	Fairness	Info. Scope	Coordination	Priority-Adaptive
<i>Centralized Approaches</i>						
Optimization	[8]–[11]	✓	✗	Global	Centralized	✗
Fair Optimization	[7], [19], [20]	✓	✓	Global	Centralized	✗
AoI-aware Scheduling	[24]–[26]	✓	✗	Global	Centralized	✗
Reinforcement Learning	[15]–[18]	✓	✗	Global	Centralized	✗
<i>Distributed Approaches</i>						
Game Theory	[12]–[14]	✓	✗	Neighbor	Iterative	✗
Consensus Optimization	[21]	✓	✓	Neighbor	Consensus	✗
Random Access	[27]	✓	✗	Local	None	✗
Priority-aware MAC	[28]	✓	✗	Local	None	✗
AoI-aware Random Access	[29]	✓	✗	Local	None	✗
Distributed AoI Scheduling	[30]	✓	✗	Global	Broadcast	✗
<i>Our Approaches</i>						
Statistical Backoff	DFF-PTVC	✓	✓	Local	Broadcast	✓
Deep Q-Learning	DFF-PTVC-DRL	✓	✓	Local	Broadcast	✓

Info. Scope: Information required for scheduling decisions. Global: network-wide state/priority; Neighbor: neighboring nodes' information; Local: only local information. **Coordination:** Communication pattern for coordination. Centralized: central controller collects and distributes information; Iterative: game-theoretic neighbor-to-neighbor exchange to reach Nash equilibrium; Consensus: consensus-based optimization through neighbor-to-neighbor variable exchange; Broadcast: node broadcasts information, no inter-node exchange. **Priority-Adaptive:** Whether the method can adapt to time-varying node priorities without requiring re-optimization or recollection of global priority information.

theoretical analysis with rigorous proofs for message scheduling can be presented based on the number of communication rounds and slots. Each node is equipped with a half-duplex transceiver; that is, during each time slot, a node can either transmit or listen, but not both. The communication of nodes in a slot is synchronized. We also assume reliable short acknowledgement feedback considered in previous work [30], [32], which can be supported by mature wireless mechanisms such as acknowledgement, retransmission, channel hopping, redundant communication, and URLLC-oriented short-packet transmission [33], [34]. We refer to nodes that transmit as transmitters, and those that listen as listeners.

In a wireless channel, the strength of a signal attenuates as the transmission distance increases. Meanwhile, the signals from multiple transmitters accumulate at the receiver. To capture the attenuation and accumulation characteristics of wireless signals, we adopt the following SINR (Signal-to-Interference-plus-Noise Ratio) communication model.

$$\begin{aligned} \text{Signal}(u, v) &= \mathcal{P}_u / d(u, v)^\alpha, \\ \text{SINR}(u, v) &= \frac{\text{Signal}(u, v)}{\sum_{w \in S/\{u\}} \text{Signal}(w, v) + N}. \end{aligned} \quad (1)$$

In the above equation, $\text{Signal}(u, v)$ is the strength of the signal at v from u . $\mathcal{P}_u$ is the transmission power of u , $d(u, v)$ is the Euclidean distance between u and v , and $\alpha \in (2, 6]$ is a path-loss exponent determined by the medium and some other factors in the environment. As the distance between u and v increases, $\text{Signal}(u, v)$ decreases accordingly. When v is trying to decode the signal from u , $\sum_{w \in S/\{u\}} \text{Signal}(w, v)$ is the interference, in which S is the set of nodes simultaneously transmitting with u , and N is the ambient noise determined by the environment. When $\text{SINR}(u, v) \geq \beta$, we say that the

transmission from u can be decoded by v , where $\beta > 1$ is a threshold determined by the hardware. The SINR model has been widely used in Internet-of-Things networks [35], [36].

Message Generation. In each round t , end node v_i collects environmental information and estimates its importance via a filter [37]. If deemed important, a message $\mathcal{M}_i^t$ is generated with priority $w_i^t \in [2, K]$ proportional to its importance and stored in queue Q_i . If Q_i is empty, a heartbeat message with weight 1 is generated instead, allowing the server to detect node crashes from prolonged silence. We adopt the stochastic arrival model of [30], where message arrivals are unknown and unpredictable, with the sole constraint that the expected number of important messages arriving per round is at most $\zeta < 1$, referred to as the message generation ratio. As shown in [38], $\zeta \leq 1$ is a necessary condition for any communication algorithm over a wireless channel where the server receives at most one message per round.

Problem Definitions. For any message $\mathcal{M}_i^t$ generated by the end node v_i at communication round t and received by the server at communication round t' , we define its *latency* as $\mathcal{L}(\mathcal{M}_i^t) = t' - t$. This latency metric captures the scheduling delay—how many rounds a message waits before being successfully transmitted and received.

The objective of our weighted information scheduling is to *minimize the maximum expected weighted latency* (MMEWL) over all messages in an interval I , i.e.,

$$\min \left(\max_{\mathcal{M}_i^t \in M(I)} \{w_i^t \cdot \mathbb{E}[\mathcal{L}(\mathcal{M}_i^t)]\} \right), \quad (2)$$

where $M(I)$ denotes the set of messages that arrived at the end nodes during the interval I . Compared with minimizing the averaged value, minimizing the maximum in the MMEWL problem provides a worst-case latency guarantee for messages

upon their reception by the edge server. In this paper, we pursue this goal through a distributed design that combines weight-proportional scheduling to balance weighted service opportunities and adaptive contention control to reduce delay caused by under-utilization or excessive collisions.

Capability and Knowledge of Nodes. We assume that all nodes use a uniform transmission power $\mathcal{P}$, a configuration referred to as the uniform power setting in prior work [32]. To satisfy the single-hop assumption, we assume that for any pair of nodes u and v in the set $V \cup v_e$, it holds that $\frac{\mathcal{P}}{d(u,v)^\alpha} \geq N\beta$. Only the edge server is equipped with a physical carrier sensing function that allows it to measure the signal strength in the wireless channel. Each node is only aware of the priorities of its own messages, but not those of other nodes. Unless otherwise stated, our analysis follows the standard assumption that participating nodes execute the protocol honestly. Adversarial behaviors such as malicious weight inflation or Byzantine attacks are outside the formal system model considered in this paper. The number of end nodes n , the SINR parameters α and β , and the ambient noise N are used only in the theoretical analysis and are not required by the end nodes or the edge server in the practical algorithm implementation.

IV. DISTRIBUTED SCHEDULING ALGORITHM

A. High-level Description for our Solution

Designing an efficient and fair scheduling algorithm for weighted messages is not an easy task in a distributed setting, especially when the weights of messages and nodes are kept as private information. An intuitive approach to fairness is to have the edge server v_e broadcast a meta probability p_e . Then, each message $\mathcal{M}_i^t$ gets scheduled with a probability $w_i^t \cdot p_e$. Here, w_i^t is the weight of the message $\mathcal{M}_i^t$. By doing this, the fairness of scheduling can be guaranteed.

However, how to choose an appropriate meta probability remains a critical issue. An excessively large meta probability may result in too many messages being transmitted simultaneously, leading to scheduling failures, commonly referred to as collisions in a physical interference channel. Conversely, if the meta probability is too small, it is likely that no scheduling occurs in the current round, reducing scheduling efficiency. To handle this issue, an adaptive scheme is implemented at the edge side, which updates the meta probability based on channel usage. Specifically, with physical carrier sensing, the edge server can know the usage of the channel in the previous round [32], i.e., the channel is idle with no messages scheduled, or is busy with too many messages transmitted. Therefore, the adaptive scheme adjusts the meta probability based on the previous channel usage, improving scheduling efficiency even when the number and weights of messages fluctuate due to messages generated by end nodes or received by the server. In Sections IV and V, we introduce an adaptive scheme based on the SEB (Statistical Exponential Back-off) method and provide a rigorous analysis demonstrating its asymptotically optimal efficiency. Later, in Section VI, the adaptive scheme is developed based on a DRL (Deep Reinforcement Learning) method, which can achieve better performance than that with the

Algorithm 1: Fast and Fair PTVC Algorithm

Initialization for the edge server v_e :
1 Meta probability $p_e \leftarrow 1/K$; $\mathcal{M}_e \leftarrow \phi$;
In each communication round, v_e does:
2 Message $\mathcal{M}_e \leftarrow \{p_e\}$;
3 Transmit $\mathcal{M}_e$; // in slot 1
4 Listen to sense the channel; // in slot 2
5 **if** receive a message $\mathcal{M}_j$ from node v_j in slot 2 **then**
6 Message $\mathcal{M}_e \leftarrow \{ACK_j\}$;
7 Transmit $\mathcal{M}_e$; // in slot 3
8 **if** the channel sensed in slot 2 is idle **then**
9 $p_e \leftarrow \min\{2p_e, 1/K\}$;
if the channel sensed in slot 2 is busy **then**
10 $p_e \leftarrow p_e/2$;

Initialization for each end node $v_i \in V$:
11 $p_i \leftarrow 0$; $Q_i \leftarrow \phi$; $w_i \leftarrow 0$;
In each communication round, node v_i does:
12 Listen to channel; // in slot 1
13 **if** receive $\mathcal{M}_e = \{p_e\}$ **then**
14 $p_i = \min\{w_i \cdot p_e, 1/2\}$;
15 $\mathcal{M}_i \leftarrow \mathcal{M}_i^t \in Q_i$ with probability of w_i^t/w_i ;
16 Transmit $\mathcal{M}_i$ with probability of p_i ; // in slot 2
17 Listen to the channel; // in slot 3
18 **if** receive an ACK_j and $j = i$ **then**
19 $Q_i \leftarrow Q_i \setminus \{\mathcal{M}_i^t\}$; // $\mathcal{M}_i^t$ is the
 message transmitted in slot 2
20 $w_i \leftarrow w_i - w_i^t$;
21 **if** message $\mathcal{M}_i^{t'}$ arrived at the node v_i in this round
 with weight $w_i^{t'}$ **then**
22 $Q_i \leftarrow Q_i \cup \{\mathcal{M}_i^{t'}\}$;
23 $w_i \leftarrow w_i + w_i^{t'}$;
24 **if** $Q_i = \phi$ **then**
 $Q_i \leftarrow \{\mathcal{M}_i^{t+1}\}$; $w_i \leftarrow 1$;

SEB scheme. Since the DRL scheme requires non-negligible energy and computational resources at the edge server, the first scheme is designed for battery-powered scenarios, whereas the second scheme is appropriate for edge devices with ample energy and computing capacity.

B. Detailed Description for Algorithm Design

The execution time of our algorithm is divided into successive communication rounds, each consisting of three slots. In the first slot of each round, the edge server broadcasts the meta probability to all the end nodes; in the second slot, each end node tries to transmit a message from its queue based on the message weights and the received meta probability; in the third slot, the edge server sends an acknowledgement if a message is received in slot 2. Each successfully acknowledged message is removed from the queue by its corresponding end node. At the end of each round, the edge server updates its meta probability with an adaptive scheme. The detailed description of our distributed scheduling algorithm on the edge and the

end side is given in the following, with pseudocode given in Algorithm 1.

On the edge side, the initial meta probability is set to $p_e = 1/K$. Then, in the first slot of each round, the edge server v_e transmits the meta probability with power P . Since v_e is the only transmitter in the current slot, all end nodes can successfully receive the meta probability. In the second slot, v_e listens to receive messages from end nodes and to sense the channel. In the third slot, if v_e has received a message $\mathcal{M}_j$ from an end node v_j , it broadcasts an acknowledgement message ACK_j . At the end of each round, the edge server updates its meta probability using the following adaptive strategy: (1) if the channel sensed in slot 2 is idle, i.e., no transmission occurs in slot 2, the meta probability is doubled, but capped at $1/K$; (2) if the channel sensed in slot 2 is busy, i.e., more than one end node transmits in slot 2, the meta probability is halved; (3) otherwise, the meta probability remains unchanged. In case (3), only one end node transmits in slot 2 and the edge server receives its message. Since the above adaptive scheme involves only simple computations, no additional slot is required in our algorithm design. The main deployment requirement is slot-level synchronization, which is standard in practical wireless systems, and the slot-1 broadcast can naturally serve as the timing reference. Therefore, the edge server only coordinates channel contention through the common meta probability p_e and channel sensing, rather than collecting or processing individual message priorities.

Each end node v_i is initialized with weighted probability $p_i = 0$, queue $Q_i = \phi$, and weight $w_i = 0$. In each round, the weight of node v_i is the sum of the weights of all messages in the queue Q_i . In slot 1, the end node v_i listens to receive the meta probability p_e from the edge server. Then, the weighted probability p_i of node v_i in this round is set to $w_i \cdot p_e$. In slot 2, if the queue Q_i is non-empty, a message $\mathcal{M}_i \leftarrow \mathcal{M}_i^t \in Q_i$ is selected with probability w_i^t/w_i . That is, exactly one message is selected from queue Q_i , and each message $\mathcal{M}_i^t$ in the queue is selected with probability proportional to its weight, so that higher-weight messages are more likely to be chosen while ensuring that all messages remain eligible for selection. In this way, the handling of message priorities is fully local: each end node uses only its own queue weights and the broadcast meta probability, without revealing individual priority information to the edge server or other end nodes. Node v_i then transmits the selected message $\mathcal{M}_i$ with probability p_i in slot 2. In slot 3, the end node v_i listens. If it receives an acknowledgement message ACK_i , the transmitted message $\mathcal{M}_i = \mathcal{M}_i^t$ is removed from queue Q_i , and the weight is updated by setting $w_i \leftarrow w_i - w_i^t$. Otherwise, if no acknowledgement is received (either because the message was not transmitted or because of a collision), the message remains in Q_i for subsequent transmissions. Finally, at the end of the current round, for each message $\mathcal{M}_i^{t'}$ that arrived at node v_i in the current round, it is added to the queue Q_i , and the weight of node v_i is updated as $w_i \leftarrow w_i + w_i^{t'}$.

In our algorithm, the adaptive strategy helps the edge server determine an appropriate meta probability for efficient communication, even as the number and weight of messages on the end side change in each round. After receiving the meta prob-

ability, each message $\mathcal{M}_i^t$ is scheduled with probability $w_i^t \cdot p_e$, which ensures weighted fairness because messages with higher priorities obtain proportionally higher scheduling probabilities, as formally established in Theorem 1. Additionally, the weight of each message is protected as private information, known only by its corresponding end node. Theorem 1 summarizes the performance of our algorithm and is proved in Section V.

Theorem 1. *We consider a single-hop wireless network with n end nodes and an edge server, in which the important messages are generated by the end nodes with some constant ratio $\zeta > 0$ and have their weights varying within $[2, K]$. By executing our algorithm for a sufficiently long time, the following properties can be guaranteed:*

- 1) **Weighted Fairness:** let $\hat{Pr}[\mathcal{M}_i^t]$ be the probability that a message $\mathcal{M}_i^t$ gets received by the edge server. Then, for any two messages $\mathcal{M}_i^t$ and $\mathcal{M}_j^{t'}$ in the same round, we have $\hat{Pr}[\mathcal{M}_i^t] : \hat{Pr}[\mathcal{M}_j^{t'}] = \Theta(w_i^t : w_j^{t'})$;
- 2) **High Throughput:** the number of messages received by the edge server in each round is $\Omega(1)$, which is asymptotically optimal;
- 3) **Low Latency:** the expected latency for a message with weight of k is $O(\frac{Kn}{k})$ rounds, and the expected maximum weighted latency for all messages is $O(Kn)$ rounds;

Corollary 1. *Under the conditions of Theorem 1, the number of messages that have been generated by the end nodes but not yet received by the edge server is bounded by $O(Kn)$.*

Proof. We prove this by contradiction. Suppose the number of undelivered messages is $\omega(Kn)$. According to the SINR model, at most one message can be successfully received by the edge server in each round. Therefore, the message with weight $k = 1$ must wait for at least $\omega(Kn)$ rounds. However, Theorem 1 guarantees that the expected latency for a message with weight $k = 1$ is $O(Kn/1) = O(Kn)$ rounds. Since $\omega(Kn)$ grows faster than any constant multiple of Kn , this contradicts the $O(Kn)$ upper bound. Therefore, the number of undelivered messages must be bounded by $O(Kn)$. ■

Discussion for the Message Generation Ratio ζ . According to the SINR Equation, the edge server at most receives 1 message in each communication round. Thus, the maximum value of ζ is no larger than 1. Otherwise, such a problem investigated in this paper is unfeasible since more and more messages will be delayed on the end side (generated by the end nodes but not received by the edge server) with their latency increased. In Theorem 1, ζ is a positive constant with an upper bound smaller than 1. The upper bound of ζ is determined by the value of the hyperparameter in our algorithm. In this paper, we prove that the maximum message generation ratio ζ that can be addressed by our scheduling algorithm is at least a positive constant, which is asymptotically optimal. In simulation, we show the performance of our algorithm with various ζ and investigate the maximum ζ that our algorithm can tolerate. Additionally, we show that the throughput of our algorithm (i.e. the average number of messages that can be

received by the server in each round) is also a positive constant in expectation, which is also asymptotically optimal.

V. ALGORITHM ANALYSIS

In this section, we demonstrate how our algorithm achieves high throughput and weighted fairness through a step-by-step analysis, and ultimately prove Theorem 1. Our analysis relies on standard mathematical tools including asymptotic notation (O , Ω , Θ) and Chernoff bounds. These preliminaries are provided in Appendix A for reference.

A. Proof for Weighted Fairness in Theorem 1

Define $Pr[\mathcal{M}_i^t]$ and $\hat{P}r[\mathcal{M}_i^t]$ as the probabilities that a message $\mathcal{M}_i^t$ gets scheduled on the end side and received by the edge server. In each communication round of our scheduling algorithm, the edge server broadcasts a meta probability p_e to all the end nodes. Each end node v_i selects a message $\mathcal{M}_i^t$ from its queue with a probability w_i^t/w_i , and transmits the message with a probability p_i . Note that w_i^t is the weight of message $\mathcal{M}_i^t$, w_i is the weight of the end node v_i with $w_i = \sum_{\mathcal{M}_i^t \in Q_i} w_i^t$ and $p_i = p_e \cdot w_i$. Therefore, the event that message $\mathcal{M}_i^t$ is scheduled by node v_i corresponds to the message being selected from Q_i and broadcast by v_i . That is:

$$Pr[\mathcal{M}_i^t] = \frac{w_i^t}{w_i} \cdot p_i = \frac{w_i^t}{w_i} \cdot w_i \cdot p_e = w_i^t \cdot p_e, \quad (3)$$

it follows that $Pr[\mathcal{M}_i^t] : Pr[\mathcal{M}_j^{t'}] = w_i^t : w_j^{t'}$ for any two messages $\mathcal{M}_i^t$ and $\mathcal{M}_j^{t'}$ in the same round.

As for the event that a message $\mathcal{M}_i^t$ is received by the edge server, it is a more complex case since the contention between simultaneous transmissions should be considered. In this section, we simplify the transmissions into the following three cases: (a) no transmission occurs and the channel is idle; (b) only one transmission occurs and it succeeds; (c) more than one transmission occurs and all of them fail because of the high contention. This simplification is a conservative assumption commonly used in wireless network analysis [30], [39], [40]. While the SINR model allows for successful concurrent transmissions when nodes are sufficiently separated spatially, our assumption provides a *lower bound* on the success probability, ensuring our analysis yields provable performance guarantees. Then, for a message $\mathcal{M}_i^t$ on node v_i ,

$$\hat{P}r[\mathcal{M}_i^t] = Pr[\mathcal{M}_i^t] \cdot \Pi_{v_j \in V \setminus v_i} (1 - p_j). \quad (4)$$

Thus, we have $\hat{P}r[\mathcal{M}_i^t] : \hat{P}r[\mathcal{M}_j^{t'}] = Pr[\mathcal{M}_i^t] : Pr[\mathcal{M}_j^{t'}] = w_i^t : w_j^{t'}$ for any two messages $\mathcal{M}_i^t$ and $\mathcal{M}_j^{t'}$ on the same node v_i ; and $\hat{P}r[\mathcal{M}_i^t] : \hat{P}r[\mathcal{M}_j^{t'}] = Pr[\mathcal{M}_i^t] \cdot (1 - p_j) : Pr[\mathcal{M}_j^{t'}] \cdot (1 - p_i) = \Theta(w_i^t : w_j^{t'})$ for any two messages $\mathcal{M}_i^t$ on end node v_i and $\mathcal{M}_j^{t'}$ on end node v_j . Since $p_i \leq 1/2$ by line 14 in Algorithm 1, we have $1 - p_i \geq 1/2$ for any node v_i , thus $(1 - p_i)/(1 - p_j) = \Theta(1)$, which implies $\hat{P}r[\mathcal{M}_i^t]/\hat{P}r[\mathcal{M}_j^{t'}] = \Theta(w_i^t/w_j^{t'})$, or equivalently, $\hat{P}r[\mathcal{M}_i^t] : \hat{P}r[\mathcal{M}_j^{t'}] = \Theta(w_i^t : w_j^{t'})$.

B. Proof for High Throughput in Theorem 1

In each communication round of our algorithm, the edge server broadcasts a meta probability p_e , and all the messages $\mathcal{M}_i^t$ with weight w_i^t have the probability $\min\{p_e \cdot w_i^t, 1/2\}$ to be scheduled. Since the efficiency of information scheduling highly relies on the meta probability, we prove the efficiency of our algorithm on scheduling by showing (1) which level of meta probability is efficient for information scheduling in Lemma 2, and (2) the adaptive scheme is powerful on choosing the appropriate meta probability by Lemma 3.

Lemma 2. When $p_e = \Theta(\frac{1}{\sum_{v_i \in V} w_i})$ in each round, the edge server receives a message with a constant probability.

Proof. Let $\mathcal{E}_1$ be the event that only one end node transmits in slot 2 of a communication round, and $Pr[\mathcal{E}_1]$ be the corresponding probability. Then, according to the SINR model, $\mathcal{E}_1$ is a sufficient condition for the edge server to receive a message, and we have:

$$\begin{aligned} Pr[\mathcal{E}_1] &= \sum_{v_i \in V} p_i \Pi_{v_j \in V \setminus v_i} (1 - p_j) \\ &= \Pi_{v_j \in V} (1 - p_j) \cdot \sum_{v_i \in V} \frac{p_i}{1 - p_i} \\ &\geq \left(\frac{1}{4}\right)^{\sum_{v_j \in V} p_j} \cdot \sum_{v_i \in V} p_i \\ &\geq \left(\frac{1}{4}\right)^{\sum_{v_j \in V} w_j \cdot p_e} \cdot \sum_{v_i \in V} w_i \cdot p_e \\ &= \Theta(1), \end{aligned} \quad (5)$$

in which the inequality $(1/4)^{\sum_{v_i \in V} p_i} \leq \Pi_{v_i \in V} (1 - p_i) \leq (1/e)^{\sum_{v_i \in V} p_i}$ for any $p_i \in (0, 1)$ is used. ■

Following a similar definition in [32], we say $p_e = \Theta(\frac{1}{\sum_{v \in V} w_v})$ is a *good* meta probability for information scheduling, and we say a round is *good* if $p_e = \Theta(\frac{1}{\sum_{v \in V} w_v})$. Then, we bound the number of *good* rounds in an interval I consisting of $\Omega(\log_2 n)$ rounds.

Lemma 3. For any sufficiently large interval I consisting of $\Omega(\log_2 n)$ rounds, at least a constant fraction of rounds in I are good with high probability.

Proof. Let X_0, X_1, X_2 be the number of rounds in which $p_e = o(\frac{1}{\sum_{v_i \in V} w_i})$, $p_e = \Theta(\frac{1}{\sum_{v_i \in V} w_i})$, $p_e = \omega(\frac{1}{\sum_{v_i \in V} w_i})$, and $\mathbb{E}(X_0), \mathbb{E}(X_1), \mathbb{E}(X_2)$ be their values in expectation, respectively. I is the number of rounds in the whole interval. Our adaptive scheme is based on a statistical exponential back-off method, in which the meta probability doubles when the channel is idle and reduces by half when the channel is busy. From a statistical perspective, when $p_e = o(\frac{1}{\sum_{v \in V} w_v})$, the channel is more likely to be idle in slot 2 of the current round since the expected number

of transmitters is $o(1)$. Then, the server doubles the meta probability at the end of the current round. Conversely, when $p_e = \omega(\frac{1}{\sum_{v \in V} w_v})$, the meta probability is more likely to decrease, as the channel tends to be busy. With such a statistical exponentially back-off method, we can prove that $\mathbb{E}(X_1)/|I|$ is a constant by the following claim.

Claim 1. With high probability, $\mathbb{E}(X_0) \leq c_1 \cdot |I|$ and $\mathbb{E}(X_2) \leq c_2 \cdot |I|$, where we can take $c_1 = 1/3$ and $c_2 = 1/3$, so that $c_1 + c_2 < 1$.

Since $X_0 + X_1 + X_2 = |I|$, we prove that $\mathbb{E}(X_1) = \Theta(|I|)$. By taking a Chernoff bound, we obtain the result that $X_1 \geq \mathbb{E}(X_1)/2$ with high probability in interval I . ■

The detailed proof of Claim 1, which establishes bounds on $\mathbb{E}(X_0)$ and $\mathbb{E}(X_2)$ through a series of technical lemmas analyzing the adaptive behavior of our algorithm, is provided in Appendix B. In summary, from Lemma 2, Lemma 3 and Proof for Claim 1, we can see that: (1) for any interval I consisting of $\Omega(\log_2 n)$ rounds, at least a constant fraction of rounds in I are *good* with high probability; (2) in a *good* round, a message can be successfully received by the edge server with a constant probability. More precisely, let $c_{\text{good}} > 0$ denote the constant fraction of good rounds guaranteed by Lemma 3, and let $c_{\text{succ}} > 0$ denote the constant success probability in each good round guaranteed by Lemma 2. Then the expected number of messages received per round is at least $c_{\text{good}} \cdot c_{\text{succ}} = \Omega(1)$, which proves that the throughput of our algorithm is $\Omega(1)$. According to the SINR Equation, the edge server at most receives 1 message in each round. Thus, the throughput of our algorithm is asymptotically optimal.

C. Proof for Low Latency in Theorem 1

Lemma 4. The expected latency for a message with weight k is $O(Kn/k)$.

Proof. We prove that a message $\mathcal{M}$ with weight k will be received by the edge server within $\Theta(Kn/k)$ rounds in expectation after it is generated by the end nodes. Let $\mathcal{E}_2$ be the event that the edge server receives a message in round t , and $\mathcal{E}_3$ be the event that the message $\mathcal{M}$ with weight k is received by the server in round t . $\Pr[\mathcal{E}_2]$ and $\Pr[\mathcal{E}_3]$ are the corresponding probabilities. According to the weighted fairness (Theorem 1 item 1) and constant throughput (Theorem 1 item 2), we obtain the following two results with high probability: (1) $\Pr[\mathcal{E}_3|\mathcal{E}_2] = \Theta(\frac{k}{\sum_{v_i \in V} w_i})$, and (2) the average probability $\Pr[\mathcal{E}_2]$ in each round is a positive constant. Thus, $\Pr[\mathcal{E}_3] = \Pr[\mathcal{E}_3|\mathcal{E}_2] \cdot \Pr[\mathcal{E}_2] = \Theta(\frac{k}{\sum_{v \in V} w_v})$. Besides, when the parameter ζ in stochastic arrival mode is smaller than the throughput proved in Part B, the number of messages generated by the end nodes but not received by the edge server has an upper bound $O(n)$, which can be proved with a similar analysis in [38]. Since the weight of each message is within the range $[1, K]$, the sum of weights for all messages on the end side, which is also the sum of weights for all end nodes, is bounded by $O(Kn)$ in each round. Thus, we have $\Pr[\mathcal{E}_3] = \Omega(\frac{k}{Kn})$ and the expected number of rounds for message $\mathcal{M}$ to be received by the server is $O(\frac{Kn}{k})$. Therefore,

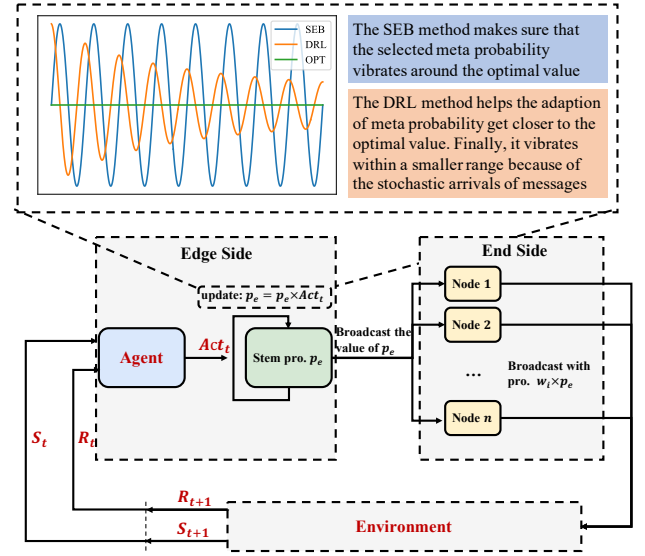


Fig. 2: The flowchart of our DRL-based approach

the expected latency for a message with weight k is $O(Kn/k)$ rounds, and the maximum expected weighted latency in our edge network is $O(\frac{Kn}{k} \cdot k) = O(Kn)$. ■

VI. DEEP REINFORCEMENT LEARNING BASED APPROACH

To achieve finer-grained control over p_e , we introduce a Deep Reinforcement Learning (DRL) based scheme complementing the SEB mechanism. Unlike SEB, which uses predefined multiplicative factors, the DRL agent learns complex mappings from network state to optimal p_e adjustments, enabling faster convergence and better adaptation to dynamic conditions. The agent is deployed at the edge server: each round, it observes the network state, selects an action to scale p_e , and broadcasts the updated value to all end nodes, which then transmit with probability $w_i \cdot p_e$. The agent receives a reward based on transmission outcomes and updates its policy accordingly (Figure 2). We employ a Double Deep Q-Network [41] to stabilize training and mitigate Q-value overestimation.

State Space. In slot 2 of each communication round t , the agent observes a seven-dimensional state vector $S_t = [U_s, Sig, SR, CR, p_e, w_{tx}, DT]$. The channel usage indicator U_s reflects the transmission outcome, taking values of 0 for an idle channel, 1 for a successful transmission, and 2 for a collision. The signal strength Sig represents the normalized total signal power detected by the edge server. The success rate SR and collision rate CR are computed over a sliding window of the most recent rounds (20 rounds in our simulation), providing adaptive indicators of recent channel performance. The current meta probability p_e represents the agent's policy parameter. Crucially, w_{tx} denotes the weight of the transmitted message in the current round if a transmission occurs (otherwise set to 0), allowing the agent to implicitly consider message priorities when adjusting p_e . Finally, the delay trend DT indicates whether weighted message delays are increasing

or decreasing, computed by comparing recent successful transmissions to earlier ones within a sliding window. These state features are chosen to capture the current channel condition, the recent operating trend, and the fairness-related effect of the current policy, using only information directly observable at the edge server. This observable-only design ensures that all state features can be obtained by the base station without requiring global queue knowledge. Accordingly, we do not use quantities such as per-node queue length or total waiting time, because they would require additional queue reporting from end nodes; instead, the delay trend DT serves as an observable proxy for recent latency evolution.

Action Space. The agent selects an action Act_t from a discrete set of 50 scaling factors uniformly distributed in $[0.1, 5.0]$ with step size 0.1. The selected action is applied as $p_e \leftarrow Act_t \cdot p_e$, enabling both conservative ($Act_t < 1$) and aggressive ($Act_t > 1$) meta probability adjustments.

Reward. The reward function comprises four components: base reward (40%), throughput reward (20%), collision penalty (20%), and strategy reward (20%). The base reward reflects transmission outcomes with priority-weighted bonuses. The throughput reward and collision penalty use sliding-window success and collision rates as proxies for system performance. The strategy reward encourages adaptive behavior aligned with channel conditions. See Appendix C for details.

Network Architecture and Training. We employ a Double Deep Q-Network (DDQN) [41] with two fully connected networks. The Q-network is trained via the Adam optimizer with learning rate $\alpha = 0.001$ and discount factor $\gamma = 0.98$. Training employs a dual-layer mechanism: mini-training every 5 rounds (10 iterations) and major training every 15 rounds (50 iterations). The target network is synchronized every 50 training steps. An ϵ -greedy exploration strategy is employed, with ϵ decaying exponentially from 0.3 to 0.02. The replay buffer capacity is 5000 transitions, and the mini-batch size is 64. See Appendix C for complete specifications. The agent is trained online during system operation, with no offline pre-training required.

VII. SIMULATION RESULTS

This section evaluates DFF-PTVC and DFF-PTVC-DRL under varying numbers of end nodes n and message arrival rates ζ . We consider the maximum and average weighted message latency to characterize worst-case and average scheduling performance, respectively. The proposed algorithms are compared with six baselines: OPT, SEB [32], Slotted ALOHA [27], ADRA [29], EDCA-style weighted backoff [28], and D-IOT [30], covering centralized and representative distributed scheduling schemes. The DRL convergence results are provided in Appendix D.

A. Simulation Settings

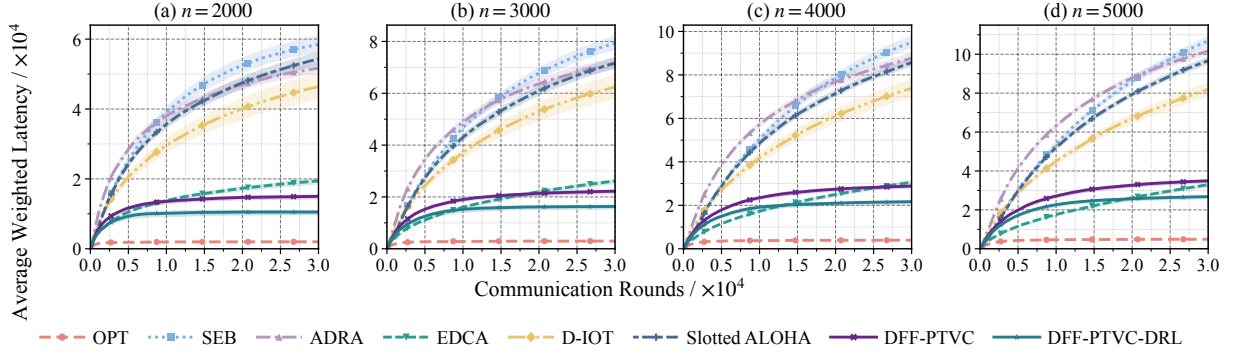
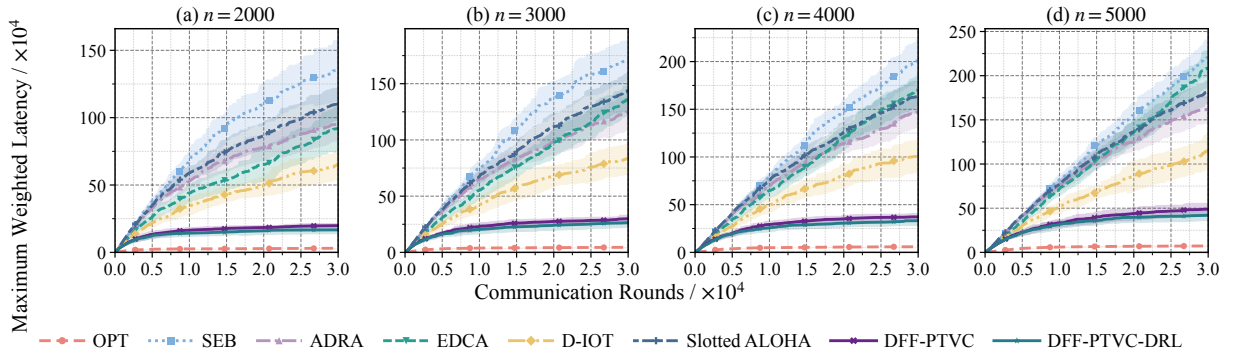
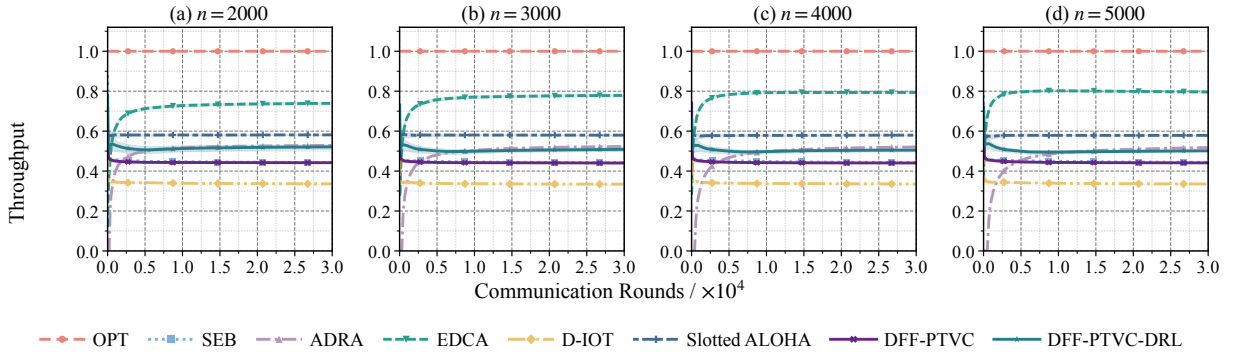
We simulate n end nodes uniformly distributed in a circular area of radius $R = 300\text{m}$, with an edge server at the center; all nodes share a transmission range of 300m. Each node maintains a message queue, initially empty. Per round, each node collects environmental information, generating a message

of weight $k \in [2, K]$ with probability ζ/n ; an empty queue triggers a heartbeat message of weight 1. Each run spans 30,000 rounds (each with three slots per our algorithm), and all reported results are averaged over 50 independent runs. Experiments ran on a Linux machine with an AMD Ryzen Threadripper PRO 5995WX (64-core), 256 GB RAM, and dual NVIDIA RTX 4090 GPUs, using Python 3.10. The SINR parameters, the priority bound K , and the access rules and parameters of all the compared methods are reported in Appendix E. All methods replay the same node placement and the same message arrivals in each run, so the comparison is made under identical conditions.

B. Performance Comparison

We empirically evaluate our proposed DFF-PTVC and DFF-PTVC-DRL algorithms against baselines: a centralized optimal solution (OPT), a distributed method using a Statistical Exponential Back-off mechanism (SEB) [32], Slotted ALOHA [27], ADRA [29], EDCA-style weighted back-off [28], and D-IOT [30]. The OPT method represents a theoretical upper bound on performance, employing a greedy strategy that schedules the message with the highest current weight in a centralized, collision-free manner. The SEB baseline, while using a similar back-off strategy to our approach, is unaware of message weights, serving as an ablation study to highlight the importance of the fairness mechanism in DFF-PTVC. Slotted ALOHA and ADRA serve as weight-oblivious local-access baselines, EDCA-style weighted backoff represents priority-aware local contention, and D-IOT represents distributed scheduling with additional global-weight information. All methods are evaluated under the same message arrivals, SINR model, and simulation parameters. For this comparison, the message arrival rate is fixed at $\zeta = 0.3$, and we vary the number of end nodes $n \in \{2000, 3000, 4000, 5000\}$. The performance is analyzed in terms of average weighted latency by Figure 3, maximum weighted latency by Figure 4), and throughput by Figure 5. All numerical results are reported in Appendix F. From simulation results, we can draw the following conclusions:

- *Constant-Factor Approximation with Purely Local Information:* From Figures 3 and 4, both DFF-PTVC-DRL and DFF-PTVC converge quickly and remain stable across all network scales. Numerical results show that, relative to OPT, DFF-PTVC-DRL stays within $5.17\times$ - $5.44\times$ for average weighted latency and $5.26\times$ - $5.65\times$ for maximum weighted latency, while DFF-PTVC stays within $7.12\times$ - $7.50\times$ and $6.34\times$ - $6.77\times$, respectively. The near-constant approximation factors across different n empirically support the linear-in- n latency behavior predicted by our theoretical analysis.
- *Weight-Aware Scheduling is Essential:* By comparing our algorithms with SEB, Slotted ALOHA, and ADRA, we observe a consistent performance gap between weight-aware and weight-oblivious methods. Across all network sizes, these three baselines have average weighted latency between $19.67\times$ and $29.19\times$ of OPT, much higher than those of both DFF-PTVC and DFF-PTVC-DRL. This

Fig. 3: Average weighted latency when $n \in \{2000, 3000, 4000, 5000\}$ and $\zeta = 0.3$ Fig. 4: Maximum weighted latency when $n \in \{2000, 3000, 4000, 5000\}$ and $\zeta = 0.3$ Fig. 5: Throughput when $n \in \{2000, 3000, 4000, 5000\}$ and $\zeta = 0.3$

confirms that purely local random access without weight awareness is insufficient for weighted-latency scheduling.

- *Existing Weight-Aware Baselines Have Clear Trade-offs:* EDCA-style weighted backoff and D-IOT both use priority information, but neither achieves balanced performance. EDCA-style weighted backoff is competitive in average weighted latency, yet its maximum weighted latency is much larger ($28.80\times$ - $30.40\times$ of OPT), indicating poor service to low-weight messages. D-IOT achieves better maximum weighted latency than the other non-PTVC distributed baselines ($15.83\times$ - $21.42\times$ of OPT), but its throughput is limited to about 0.334 due to the overhead of global-weight collection, and its average

weighted latency remains much higher than those of our two methods. These results show that weight awareness alone is not enough; effective contention control with low coordination overhead is equally important.

- *Constant Throughput Guarantees:* Figure 5 shows that both DFF-PTVC-DRL and DFF-PTVC maintain stable throughput across all network scales. Numerical results further show that DFF-PTVC-DRL achieves throughput of 0.503-0.524, while DFF-PTVC stays at 0.440-0.442, empirically validating the constant-throughput property of our design.
- *DRL Further Improves the Rule-Based Algorithm:* Across all network scales, DFF-PTVC-DRL consistently outper-

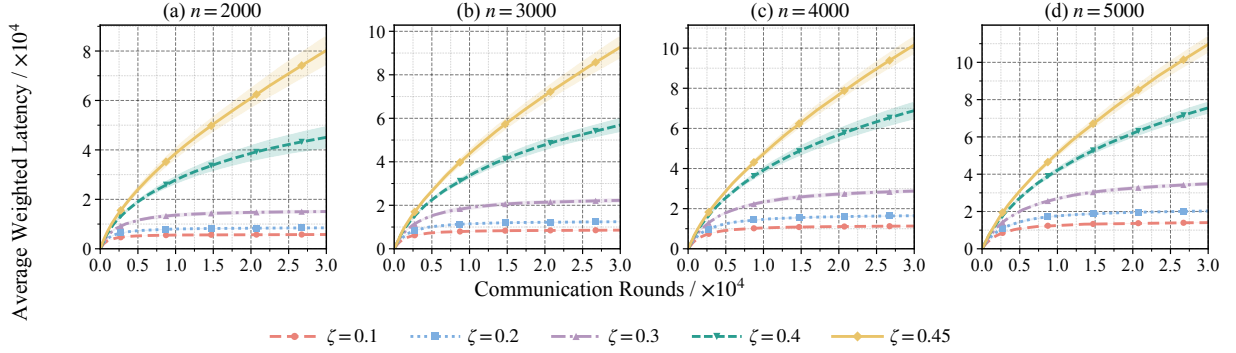


Fig. 6: Average weighted latency when $n \in \{2000, 3000, 4000, 5000\}$ and $\zeta \in \{0.1, 0.2, 0.3, 0.4, 0.45\}$

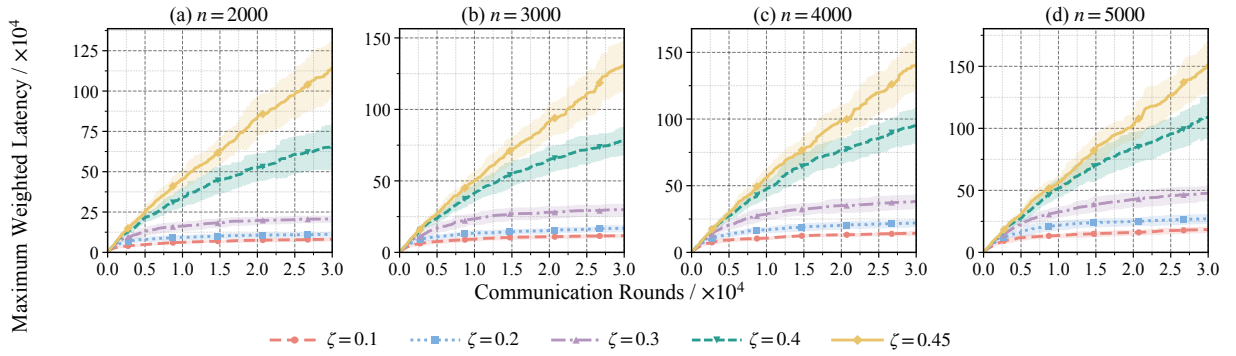


Fig. 7: Maximum weighted latency when $n \in \{2000, 3000, 4000, 5000\}$ and $\zeta \in \{0.1, 0.2, 0.3, 0.4, 0.45\}$

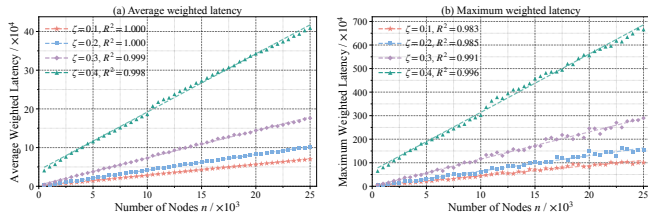


Fig. 8: Scalability of DFF-PTVC: steady-state weighted latency vs. network size n for $\zeta \in \{0.1, 0.2, 0.3, 0.4\}$. Dashed lines show linear fits with coefficients of determination R^2 .

forms DFF-PTVC. Compared with DFF-PTVC, DFF-PTVC-DRL reduces average weighted latency by 24%-31%, reduces maximum weighted latency by 15%-20%, and improves throughput by 14%-19%. This gain comes from finer-grained meta probability control than the fixed doubling/halving rule used in SEB.

C. Scalability Analysis

To investigate when DFF-PTVC can maintain stability, we test our algorithm under varying message arrival rates and network scales in Figures 6 and 7. Specifically, we first evaluate both average and maximum weighted latency by setting $\zeta \in \{0.1, 0.2, 0.3, 0.4, 0.45\}$ and $n \in \{2000, 3000, 4000, 5000\}$, where the x -axes represent the number of communication rounds and the y -axes represent the weighted latency. The

results reveal a clear stability boundary: when $\zeta \leq 0.4$, both latency metrics converge to stable values, indicating that the system can handle the load. However, when $\zeta = 0.45$, both latency curves grow continuously without converging, signaling that the arrival rate exceeds system capacity. This experimental threshold at $\zeta \in (0.4, 0.45)$ matches the measured throughput of approximately 0.44 in Figure 5, validating the queue stability conclusion from Corollary 1. For completeness, supplementary results of DFF-PTVC-DRL are provided in Appendix G, which show a similar stable convergence behavior across different network configurations.

To further evaluate the scalability of DFF-PTVC, we consider 50 network sizes from $n = 500$ to 25,000 (step size 500) under $\zeta \in \{0.1, 0.2, 0.3, 0.4\}$, with 50 independent trials per configuration. Figure 8 reports the steady-state weighted latency averaged over the final 500 communication rounds, together with linear regression results. Across all arrival rates, the R^2 values exceed 0.997 for average weighted latency and 0.983 for maximum weighted latency, indicating an approximately linear growth with network size. Appendix H further shows that the average throughput remains around 0.44 across network sizes and decreases only mildly with ζ , consistent with our $\Omega(1)$ throughput guarantee. These results agree with the $O(Kn)$ queue bound in Corollary 1 and demonstrate that DFF-PTVC scales linearly to networks with 25,000 nodes.

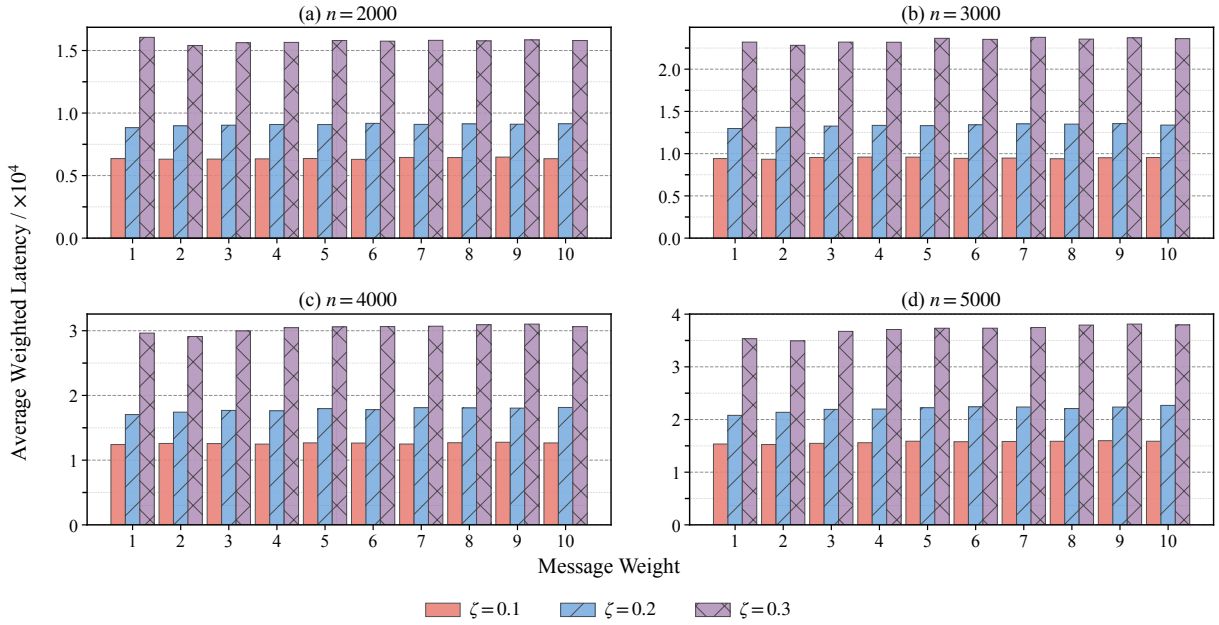


Fig. 9: Distribution of average weighted latency across different message weights

D. Weighted Fairness Analysis

In this section, we examine the weighted latency distribution across different message weights in Figure 9 and the corresponding Jain's Fairness Index in Table II. Specifically, we run DFF-PTVC for 30,000 rounds with message weights uniformly distributed in $[1, 10]$ by setting $n \in \{2000, 3000, 4000, 5000\}$ and $\zeta \in \{0.1, 0.2, 0.3\}$, where in Figure 9, the x -axes represent message weights and the y -axes represent average weighted latency. From Figure 9, we can see that the average weighted latency stays nearly constant across all weight values. For instance, at $n = 5000, \zeta = 0.1$ in Figure 9(d), all bars stabilize around 1.5×10^4 regardless of weight. This is because if our algorithm schedules messages proportionally to their weights, then a message with weight 10 should be transmitted 10 times faster than a message with weight 1, which means their weighted latency (weight times latency) should be the same. The flat distribution in Figure 9 confirms this proportional scheduling property. To quantify the fairness more rigorously, we compute Jain's Fairness Index [42], [43] in Table II, which shows that all scenarios achieve fairness scores ≥ 0.9992 , extremely close to the ideal value of 1.0 (indicating nearly perfect fairness). Moreover, by comparing the results from light load ($n = 2000, \zeta = 0.1$) to heavy load ($n = 5000, \zeta = 0.3$), we can see that the fairness index remains consistently above 0.9992 across all conditions, demonstrating that our algorithm maintains weighted fairness scheduling robustly without requiring global coordination. The weighted fairness results of DFF-PTVC-DRL are provided in Appendix G, which remain close to 1 across different traffic loads and confirm that the DRL-based edge-side adaptation does not alter the weighted fairness mechanism.

TABLE II: JAIN'S FAIRNESS INDEX OF WEIGHTED LATENCY IN DFF-PTVC WITH VARYING NETWORK SIZE (n) AND MESSAGE ARRIVAL RATE (ζ)

ζ	0.1	0.2	0.3	Std
n				
2000	0.9999	0.9999	0.9999	0.0000
3000	0.9999	0.9998	0.9998	0.0000
4000	0.9999	0.9996	0.9996	0.0001
5000	0.9998	0.9994	0.9992	0.0002
Std	0.0001	0.0002	0.0003	—

E. Simulation Summary.

In this section, we evaluate the performance and stability of both DFF-PTVC and DFF-PTVC-DRL through average weighted latency, maximum weighted latency, throughput, stability threshold, scalability, and weighted fairness under different network scales and traffic loads. The results show that DFF-PTVC-DRL achieves a constant-factor approximation of $5.17 \times - 5.65 \times$ to the centralized optimum across all network scales, while DFF-PTVC achieves $6.34 \times - 7.50 \times$, and both algorithms maintain constant throughput of approximately 0.50 and 0.44, respectively, regardless of network size, validating our $O(Kn)$ latency bound and $\Omega(1)$ throughput guarantee. The system remains stable when the arrival rate stays below the throughput capacity, with the experimental threshold at $\zeta \in (0.4, 0.45)$ matching the measured throughput. Weighted fairness is also achieved robustly, with Jain's Fairness Index of DFF-PTVC remaining above 0.9992 in the main experiments, while the supplementary results in Appendix G further show that DFF-PTVC-DRL preserves fairness and stable convergence across multiple traffic loads. Overall, these results demonstrate that our algorithms achieve near-optimal

efficiency, provable stability, and robust weighted fairness in a fully distributed manner without global priority information.

VIII. CONCLUSION

The paper investigates the distributed fast and fair physical-to-virtual communication problem for Digital Twin Edge Networks, where the challenge lies in achieving both efficiency (i.e., low latency and high throughput) and weighted fairness using only local information at each node. To handle the time-varying message priorities, a distributed algorithm DFF-PTVC is presented, in which the edge server broadcasts an adaptive meta probability while each end node makes scheduling decisions using only its local priority. With rigorous analysis, we prove that DFF-PTVC achieves asymptotically optimal throughput, low latency inversely proportional to message priority, and weighted fairness with reception probability ratios proportional to priority ratios. By extending to a deep reinforcement learning approach, we further develop DFF-PTVC-DRL for enhanced performance. Simulation results demonstrate that DFF-PTVC and DFF-PTVC-DRL achieve a constant-factor approximation of $5.17\times\text{--}7.50\times$ to the centralized optimum, and DFF-PTVC can maintain near-perfect weighted fairness. Extending our research to mobile edge networks with node mobility will be our work in the future.

REFERENCES

- [1] M. Al-Hawawreh and M. S. Hossain, "Digital twin-driven secured edge-private cloud industrial internet of things (iiot) framework," *Journal of Network and Computer Applications*, vol. 226, p. 103888, 2024.
- [2] J. N. Njoku, E. C. Nkoro, R. M. Medina, P. M. Custodio, C. I. Nwakanma, J.-M. Lee, and D.-S. Kim, "Digital twin and metaverse-enhanced battery management for electric vehicles," *High-Confidence Computing*, vol. 6, no. 1, p. 100358, 2026.
- [3] Y. Zhao, L. Zuo, D. Yu, R. Pan, L. Fu, Y. Lu, and H. Zhang, "Mobile systems for healthcare 5.0: Applications, challenges and opportunities," *High-Confidence Computing*, vol. 6, no. 2, p. 100392, 2026.
- [4] L. Yang, Y. Zou, Z. Zhang, P. Wang, D. Yu, A. Zubow, F. Dressler, and X. Cheng, "Jamming-resilient physical-to-virtual communications in digital twin edge networks," *IEEE Transactions on Networking*, 2025.
- [5] C. Gehrmann and M. Gunnarsson, "A digital twin based industrial automation and control system security architecture," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 1, pp. 669–680, 2019.
- [6] T. Jin, Z. Sun, L. Li, Q. Zhang, M. Zhu, Z. Zhang, G. Yuan, T. Chen, Y. Tian, X. Hou *et al.*, "Triboelectric nanogenerator sensors for soft robotics aiming at digital twin applications," *Nature communications*, vol. 11, no. 1, p. 5381, 2020.
- [7] J. Feng, L. Liu, X. Hou, Q. Pei, and C. Wu, "Qoe fairness resource allocation in digital twin-enabled wireless virtual reality systems," *IEEE Journal on Selected Areas in Communications*, 2023.
- [8] Z. Zhou, Z. Jia, H. Liao, W. Lu, S. Mumtaz, M. Guizani, and M. Tariq, "Secure and latency-aware digital twin assisted resource scheduling for 5g edge computing-empowered distribution grids," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 7, pp. 4933–4943, 2021.
- [9] Y. Lu, X. Huang, K. Zhang, S. Maharjan, and Y. Zhang, "Low-latency federated learning and blockchain for edge association in digital twin empowered 6g networks," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 7, pp. 5098–5107, 2020.
- [10] H. Liao, Z. Zhou, Z. Jia, Y. Shu, M. Tariq, J. Rodriguez, and V. Frascolla, "Ultra-low aoi digital twin-assisted resource allocation for multi-mode power iot in distribution grid energy management," *IEEE Journal on Selected Areas in Communications*, vol. 41, no. 10, pp. 3122–3132, 2023.
- [11] C. Ren, C. Chen, P. Li, and X. Guan, "Communication and computation resource allocation for digital twin in iiot systems," in *2021 China Automation Congress (CAC)*. IEEE, 2021, pp. 7364–7369.
- [12] J. Zheng, T. H. Luan, Y. Zhang, R. Li, Y. Hui, L. Gao, and M. Dong, "Data synchronization in vehicular digital twin network: A game theoretic approach," *IEEE Transactions on Wireless Communications*, 2023.
- [13] F. Hu, S. Bi, and Y. Zhu, "Cooperative game-based digital twin drives decision making: Overall framework, basic formalization and application case," *Mathematics*, vol. 12, no. 2, p. 355, 2024.
- [14] W. Sun, P. Wang, N. Xu, G. Wang, and Y. Zhang, "Dynamic digital twin and distributed incentives for resource allocation in aerial-assisted internet of vehicles," *IEEE Internet of Things Journal*, vol. 9, no. 8, pp. 5839–5852, 2021.
- [15] Y. Dai, K. Zhang, S. Maharjan, and Y. Zhang, "Deep reinforcement learning for stochastic computation offloading in digital twin networks," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 7, pp. 4968–4977, 2020.
- [16] Z. Zhang, Y. Huang, C. Zhang, Q. Zheng, L. Yang, and X. You, "Digital twin-enhanced deep reinforcement learning for resource management in networks slicing," *IEEE Transactions on Communications*, 2024.
- [17] K. Zhang, J. Cao, and Y. Zhang, "Adaptive digital twin and multi-agent deep reinforcement learning for vehicular edge computing and networks," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 2, pp. 1405–1413, 2021.
- [18] K. Xu, Z. Tang, H. Yu, P. Zeng, and L. Kong, "Digital twin-driven collaborative scheduling for heterogeneous task and edge-end resource via multi-agent deep reinforcement learning," *IEEE Journal on Selected Areas in Communications*, vol. 41, no. 10, pp. 3056–3069, 2023.
- [19] S. D. Okegbile, J. Cai, and A. S. Alfa, "Cost-effective mobility-aware multi-connectivity scheme for physical-to-virtual communications in digital twin networks," in *ICC 2025-IEEE International Conference on Communications*. IEEE, 2025, pp. 524–529.
- [20] X. Liu, P. Yu, H. Fang, C. Tan, F. Zhou, and W. Li, "Fair communication resource scheduling in 6g and digital twin-driven virtual power plants," in *2024 IEEE/CIC International Conference on Communications in China (ICCC Workshops)*. IEEE, 2024, pp. 271–276.
- [21] D. Van Huynh, V.-D. Nguyen, S. R. Khosravirad, and T. Q. Duong, "Fairness-aware latency minimisation in digital twin-aided edge computing with ultra-reliable and low-latency communications: A distributed optimisation approach," in *2022 56th Asilomar Conference on Signals, Systems, and Computers*. IEEE, 2022, pp. 1045–1049.
- [22] H. El-Sayed, S. Sankar, M. Prasad, D. Puthal, A. Gupta, M. Mohanty, and C.-T. Lin, "Edge of things: The big picture on the integration of edge, iot and the cloud in a distributed computing environment," *IEEE access*, vol. 6, pp. 1706–1717, 2017.
- [23] L. Yang, Y. Zou, D. Yu, and J. Yu, "Distributed age-of-information optimization in edge computing for internet of vehicles," *Journal of Systems Architecture*, vol. 144, p. 103000, 2023.
- [24] J. Li, S. Guo, W. Liang, J. Wang, Q. Chen, Z. Hong, Z. Xu, W. Xu, and B. Xiao, "Aoi-aware service provisioning in edge computing for digital twin network slicing requests," *IEEE Transactions on Mobile Computing*, vol. 23, no. 12, pp. 14 607–14 621, 2024.
- [25] Y. Guo, Y. Fu, Y. Zhang, and T. Q. Quek, "Age-of-information and energy optimization in digital twin edge networks," in *GLOBECOM 2024-2024 IEEE Global Communications Conference*. IEEE, 2024, pp. 3194–3200.
- [26] E. Bozkaya-Aras, "Digital twin-enabled age of information-aware scheduling for industrial iot edge networks," *Pervasive and Mobile Computing*, vol. 112, p. 102083, 2025.
- [27] L. G. Roberts, "Aloha packet system with and without slots and capture," *ACM SIGCOMM Computer Communication Review*, vol. 5, no. 2, pp. 28–42, 1975.
- [28] S. Banerji and R. S. Chowdhury, "On IEEE 802.11: wireless lan technology," *arXiv preprint arXiv:1307.2661*, 2013.
- [29] I. Kadota, A. Sinha, E. Uysal-Biyikoglu, R. Singh, and E. Modiano, "Scheduling policies for minimizing age of information in broadcast wireless networks," *IEEE/ACM Transactions on Networking*, vol. 26, no. 6, pp. 2637–2650, 2018.
- [30] D. Yu, Y. Zou, M. Xu, Y. Xu, Y. Zhang, B. Gong, and X. Xing, "Competitive age of information in dynamic iot networks," *IEEE Internet of Things Journal*, vol. 8, no. 20, pp. 15 160–15 169, 2020.
- [31] Y. Yigit, L. Maglaras, W. J. Buchanan, B. Canberk, H. Shin, and T. Q. Duong, "Ai-enhanced digital twin framework for cyber-resilient 6g internet-of-vehicles networks," *IEEE Internet of Things Journal*, 2024.
- [32] D. Yu, Y. Zou, Y. Zhang, H. Sheng, W. Lv, and X. Cheng, "An exact implementation of the abstract mac layer via carrier sensing in dynamic networks," *IEEE/ACM Transactions on Networking*, vol. 29, no. 3, pp. 994–1007, 2021.
- [33] A. A. Kumar S., K. Ovsthus, and L. M. Kristensen, "An industrial perspective on wireless sensor networks — a survey of requirements, protocols, and challenges," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 3, pp. 1391–1412, 2014.

- [34] G. J. Sutton, J. Zeng, R. P. Liu, W. Ni, D. N. Nguyen, B. A. Jayawickrama, X. Huang, M. Abolhasan, Z. Zhang, E. Dutkiewicz, and T. Lv, "Enabling technologies for ultra-reliable and low latency communications: From phy and mac layer perspectives," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2488–2524, 2019.
- [35] M. A. Abd-Elmagid, M. A. Kishk, and H. S. Dhillon, "Joint energy and sinr coverage in spatially clustered rf-powered iot network," *IEEE Transactions on Green Communications and Networking*, vol. 3, no. 1, pp. 132–146, 2018.
- [36] M. W. Shabir, T. N. Nguyen, J. Mirza, B. Ali, and M. A. Javed, "Transmit and reflect beamforming for max-min sinr in irs-aided mimo vehicular networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 24, no. 1, pp. 1099–1105, 2022.
- [37] M. Yuan, L. Zhang, F. He, X. Tong, and X.-Y. Li, "Infi: End-to-end learnable input filter for resource-efficient mobile-centric inference," in *Proceedings of the 28th Annual International Conference on Mobile Computing And Networking*, 2022, pp. 228–241.
- [38] D. Yu, Y. Zou, J. Yu, X. Cheng, Q.-S. Hua, H. Jin, and F. C. Lau, "Stable local broadcast in multihop wireless networks under sinr," *IEEE/ACM Transactions on Networking*, vol. 26, no. 3, pp. 1278–1291, 2018.
- [39] Y. Zou, M. Xu, J. Yu, F. Zhao, and X. Cheng, "A fast consensus for permissioned wireless blockchains," *IEEE Internet of Things Journal*, vol. 10, no. 14, pp. 12 102–12 111, 2021.
- [40] L. Yang, Y. Zou, S. Shen, P. Wang, D. Yu, and X. Cheng, "A fault-tolerant communication algorithm for age-of-information optimization in ditens," *IEEE Transactions on Communications*, vol. 72, no. 5, pp. 2851–2864, 2023.
- [41] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double q-learning," in *Proceedings of the AAAI conference on artificial intelligence*, vol. 30, no. 1, 2016.
- [42] A. B. Sediq, R. H. Gohary, R. Schoenen, and H. Yanikomeroglu, "Optimal tradeoff between sum-rate efficiency and jain's fairness index in resource allocation," *IEEE Transactions on Wireless Communications*, vol. 12, no. 7, pp. 3496–3509, 2013.
- [43] C. Guo, M. Sheng, Y. Zhang, and X. Wang, "A jain's index perspective on α -fairness resource allocation over slow fading channels," *IEEE communications letters*, vol. 17, no. 4, pp. 705–708, 2013.



Jiguo Yu received the Ph.D. degree in mathematics from Shandong University in 2004. He is currently a Professor with the University of Electronic Science and Technology of China and a joint Professor with Qilu University of Technology. His research interests include wireless networks, privacy-preserving computing, blockchain, intelligent IoT, AI and cybersecurity, distributed computing, and graph theory. He is an IEEE Fellow, an ACM member, and a CCF Senior Member.



Xiuzhen Cheng received her MS and PhD degrees in computer science from University of Minnesota, Twin Cities, in 2000 and 2002, respectively. She was a faculty member at the Department of Computer Science, The George Washington University, from 2002-2020. Currently she is a professor of computer science at Shandong University, Qingdao, China. Her research focuses on blockchain computing, security and privacy, and Internet of Things. She is a Fellow of IEEE.



Senmao Qi received the B.E. degree from the School of Computer Science and Technology, Shandong University, Qingdao, China, in 2021, and the Ph.D. degree from the School of Computer Science and Technology, Shandong University, Qingdao, China, in 2026. He is currently a Postdoctoral Researcher with Zhejiang University, Hangzhou, China. His research interests include distributed machine learning, AI security and wireless network.



Falko Dressler is a Full Professor and Chair for Data Communications and Networking at TU Berlin. He received the M.Sc. and Ph.D. degrees from the University of Erlangen in 1998 and 2003, respectively. He has served in editorial and leadership roles for major IEEE/ACM journals and conferences. He is an IEEE Fellow, ACM Distinguished Member, and member of acadtech. His research interests include adaptive wireless networking and wireless sensing for IoT and Cyber-Physical Systems.



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.



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.



Li Yang received the M.S. degree in mathematics from Fuzhou University, China, in 2020, and the Ph.D. degree in computer science from Shandong University, China, in 2024. He was a joint Ph.D. candidate at Paderborn University, Germany, from 2021 to 2022. He is currently an Assistant Professor with Qingdao University, China. His research interests include wireless networks, distributed computing, and network security.