# Matching-Theory-Based Low-Latency Scheme for Multitask Federated Learning in MEC Networks

Dawei Chen, *Student Member, IEEE*, Choong Seon Hong, *Senior Member, IEEE*, Li Wang, *Senior Member, IEEE*, Yiyong Zha, Yunfei Zhang, Xin Liu, and Zhu Han, *Fellow, IEEE*

***Abstract*—Nowadays, there is an ever-increasing interests in federated learning, which allows end devices to collaboratively train a global machine learning model in a decentralized paradigm without sharing individual data. Despite the advantages of low communication cost and preserving data privacy, federated learning is also facing with new challenges to address. Practically, end devices will consider the resources cost and willingness caused by machine learning model training when they are invited to participate a federated learning task. So, how to assign the preferable tasks to the devices with high willingness has to be considered. Besides, the end devices have the property of high mobility, which means the time of devices localizing within the network is limited. Therefore, to reduce the task execution time is necessary. To address these problems, we first analyze and formulate the latency minimization problem for multitask federated learning in a multiaccess edge computing (MEC) network scenario. Then, we model the corresponding problem as a matching game to find the optimal task assignment solutions. Moreover, considering the large-scale Internet-of-Things (IoT) scenario, it is almost impossible for two sides to know the details of every individual of the other side so that the complete preference list (CPL) cannot be built in reality. Therefore, we propose an algorithm for large-scale matching with the incomplete preference list to address the problem. Finally, we conduct the numerical simulation in various cases to demonstrate the effectiveness of our proposed method. The results show that our approach can achieve similar performance with the CPL case.**

***Index Terms*—Incomplete preference list (IPL), matching theory, multiaccess edge computing (MEC), multitask federated learning.**

Manuscript received December 5, 2020; accepted January 9, 2021. Date of publication January 21, 2021; date of current version July 7, 2021. This work was supported in part by the U.S. Multidisciplinary University Research Initiative under Grant 18RT0073; in part by NSF under Grant EARS-1839818, Grant CNS1717454, Grant CNS-1731424, and Grant CNS-1702850; in part by the National Key Research and Development Program of China under Grant 2020YFC1511801; in part by the National Natural Science Foundation of China under Grant U2066201 and Grant 61871416; and in part by the Beijing Municipal Natural Science Foundation under Grant L192030. *(Corresponding author: Li Wang.)*

Dawei Chen is with the Department of Electrical and Computer Engineering, University of Houston, Houston, TX 77004 USA (e-mail: dchen22@uh.edu).

Choong Seon Hong is with the Department of Computer Science and Engineering, Kyung Hee University, Yongin-si 17104, South Korea (e-mail: cshong@khu.ac.kr).

Li Wang is with the School of Electronic Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China, and also with the Key Laboratory of the Universal Wireless Communications, Ministry of Education, Beijing 100876, China (e-mail: liwang@bupt.edu.cn).

Yiyong Zha, Yunfei Zhang, and Xin Liu are with Tencent Technology Company Ltd., Shenzhen 518054, China (e-mail: nelsonzha@tencent.com; yanniszhang@tencent.com; xinliu@tencent.com).

Zhu Han is with the Department of Electrical and Computer Engineering, University of Houston, Houston, TX 77004 USA, and also with the Department of Computer Science and Engineering, Kyung Hee University, Seoul 446-701, South Korea (e-mail: zhan2@uh.edu).

Digital Object Identifier 10.1109/JIOT.2021.3053283

## I. Introduction

THE LAST decade has witnessed an unprecedented improvement and prosperity of machine learning techniques and applications, such as face recognition, driverless vehicles, autonomous disease diagnose, etc. On one hand, such a rapid development is heavily dependent on the tremendous available data generated by the ever-increasing number of users. According to the anticipation of International Data Corporation, the number of devices connected to the Internet will achieve 80 billions and the amount of generated data can be as much as 180 trillion GB in 2025 [1]. On the other hand, thanks to the evolution of powerful computation hardware design and efficient computing architecture, such as parallel high-performance graphics processing units (GPUs), those computation-intensive machine learning applications finally are able to be performed on devices instead of centralized cloud data centers, which also promotes the extensive use of machine learning [2].

However, it is acknowledged that training a machine learning model relies heavily on enormous data while the data generated by one single device are limited. At the same time, due to the diversity property of individual behavior characteristics, the machine learning model obtained from one device is hard to work desirably for others [3]. Besides, for traditional machine learning, the model is trained via a centralized manner, i.e., all the training data have to be uploaded to a centralized cloud through a wire or wireless channel, which increases the risk of privacy leakage [4]. Therefore, proposing a framework that can unite multiple devices to collaboratively train a universal model and guarantee the privacy safety to a certain extent simultaneously is indispensable, which motivates federated learning coming into being.

Federated learning is first proposed by Konečnỳ *et al.* [5], which is a machine learning framework that allows end devices to jointly train a global machine learning model in a decentralized paradigm without sharing individual data. Typically, there will be multiple user devices involving in a federated learning tasks. First, an initialized machine learning model will be broadcast to all of the participants. Having received the naive model, each participant will optimize the model

based on their own data through, for example, the stochastic gradient descent (SGD) method for a certain number of iterations. Then, the model updates, i.e., the calculated model parameters by each participant, will be uploaded to an aggregator. Thereupon, the aggregator performs a weighted average among all the parameters to obtain relatively optimal global model parameters, which will be fed back to all the participants again. The procedures will be conducted repetitively until the predefined accuracy is achieved.

Intuitively, there are several reasons to adopt federated learning in practical situation. First, with the development of multiaccess edge computing (MEC) networks, there will be many available edge nodes deployed at the edge, which are much closer to the user devices. Invested with adequate computing resources, the edge node can be sufficiently powerful to process federated learning tasks instead of using centralized data centers so as to reduce the transmission latency [6]. Second, since in federated learning, what transmitted between participants and aggregator are the machine learning model parameters rather than raw data, whose size is much smaller. Therefore, the communication cost can be reduced significantly [7]. At the same time, this manner can also decrease the probability of eavesdropping to a certain extend such that the privacy can be guaranteed [8]. Third, due to the diversity of individual behavior characteristics, the distributions of data generated by different devices are disparate. Fortunately, federated learning has been proved that it is effective to deal with nonidentical and independent distribution (non-i.i.d.) data [9], which is suitable for large-scale Internet-of-Things (IoT) scenarios.

With all the alluring benefits above, federated learning is also faced with new challenges to tackle. On one hand, the majority of existing literature make a desirable assumption that once the end devices are invited, they will unconditionally take part in the federated learning tasks, which is not practical in the real world. From the perspective of participants, resources cost and willingness caused by machine learning model training have to be took into consideration. For example, when the remained power is lower than a threshold, the device can be unwilling to join any tasks. Otherwise, its normal functions are not able to be supported. Second, in a MEC network, the end devices are always on the go, which means the time of devices localizing within the network is limited. Although the edge computing paradigm can reduce the latency to a certain extend, how to reduce the delay and save more time needs to be discussed further. Third, there are many available edge nodes in a MEC network, while each node can work as an agrregator actually. Therefore, how to parallelly perform multiple federated learning tasks, i.e., multitask federated learning, in a low-latency purpose to augment efficiency needs to be considered as well.

In order to address the challenges above, we propose a matching with incomplete preference list (IPL)-based method toward a low-latency purpose for multitask federated learning in the MEC network. The contribution of this work can be summarized as follows.

1) We analyze and formulate the low-latency problem for multitask federated learning in the MEC network from computation and communication perspectives. Then, the corresponding formulation is given to consider multitask federated learning. Most existing literature only consider one single federated learning problem.
2) We model the low-latency problem as the hospitals–residents (HR) matching problem. Besides, considering that the number of participants in IoT can be enormous, it is not possible for both matching sides to acknowledge the details for every individual of the other side, which means building the complete preference list (CPL) is not practical. Therefore, we propose a matching with the IPL method to solve the problem so as to get closer to reality, which is seldom done by existing literature.
3) We conduct the numerical simulations to demonstrate the effectiveness of our analysis and proposed method. Also, we discuss the influence of number of participants, the number of edge nodes, the edge node capacity, local accuracy, energy threshold, and preference list missing rate among network latency. The performance of our proposed method is close to the performance of the CPL case with small gap between them due to information missing.

The remainder of this article is organized as follows. Section II discusses some related existing literature for both federated learning and matching theory fields. Section III introduces the specific scenario, analyzes communication and computation model for federated learning, and formulates the corresponding low-latency problem for multitask federated learning in the MEC network. In Section IV, related matching definitions are given and the proposed matching algorithm with the IPL is introduced. Section V conducts the numerical simulation and discusses the results accordingly. Finally, a conclusion is drawn in Section VI.

## II. Related Work

As a promising distributed learning paradigm, federated learning has become a popular field of research. Wang *et al.* [10] analyzed the convergence bound for federated learning with non-i.i.d. data and proposed a control algorithm to achieve an optimal tradeoff between local updates and global aggregation considering a resource budget constrain. Sattler *et al.* [11] proposed a sparse ternary compression framework to reduce the communication cost for federated learning with non-i.i.d. data. Yang *et al.* [12] proposed an over-the-air computation-based approach for the fast global aggregation process so as to maximize the number of participants under limited bandwidth, which can improve the accuracy of federated learning. Kang *et al.* [13] proposed a contract theory-based method to build an incentive mechanism to motivate the participants with high-accuracy local training to take part in the collaborative learning process for efficient federated learning. Dinh *et al.* [14] proposed a fast convergence algorithm to find an optimal tradeoff between computation and communication latencies, as well as overall federated learning time and user device energy consumption, so as to enhance the performance of federated learning in wireless networks. Lim *et al.* [15] proposed

a multidimensional contract-matching incentive framework to maximization the profit of model owners in a unmanned aerial vehicle (UAV) enabled Internet of Vehicles (IoV) scenario. Zhu and Jin [16] utilized a multiobjective evolutionary algorithm to simultaneously minimize the communication cost and maximize the global model accuracy. However, regarding these works, [10]–[12] make a desirable assumption that once the end devices are invited, they will unconditionally take part in the federated learning tasks, which is not practical in the real world. Besides, for [10]–[16], only onefold federated learning task is discussed and multitask federated learning is not considered.

As for the matching theory, it is often used to address the combinatorial problem of players in two sets, based on the preferences of each player and the individual information [17]. Su *et al.* [18] proposed an algorithm that combines the Markov decision process with random serial dictatorship matching to solve the UAV-assisted charging problem for energy constrained devices. Gu *et al.* [19] proposed a student project allocation game-based matching method to address the joint radio and computation resource allocation problem for IoT in the fog computing scenario. Sharghivand *et al.* [20] proposed a two-sided matching solution for IoT and edge nodes matching problem to reduce the average service time so that Quality-of-Service (QoS) requirements can be achieved. Seng *et al.* [21] developed an efficient task-virtual machine matching algorithm that jointly considers task execution time and energy consumption to make computation offloading decisions in ultradense wireless networks. Fantacci and Picano [22] proposed a matching-based strategy for virtual machine placement so as to minimize the system response time and requests dropping for industrial IoT applications. Huang *et al.* [23] applied a matching theory-based approach to solve the computation offloading problem utilizing parked vehicles such that the more tasks can be accomplished within a certain time range. Whereas, for the above-mentioned work, the matching game is considered under the CPL situation, which is not practical in large-scale IoT applications.

## III. System Model and Problem Formulation

In this section, we introduce the preliminaries for federated learning in Section III-A. Then, the computation and communication models are discussed in Sections III-B and III-C, respectively. Finally, we describe the scenario in details and provide the corresponding formulation for multitask federated learning latency minimization problem within the MEC network in Section III-D. For a clear understanding of parameters and symbols in this article, their definitions and descriptions are provided in Table I in detail.

TABLE I
Parameter and Symbol Description

| Notation | Definition |
|---|---|
| $\mathcal{N}$ | Set of participants. |
| $\mathcal{E}$ | Set of edge nodes. |
| $N$ | Total number of participants. |
| $E$ | Total number of edge nodes. |
| $\mathcal{N}_i$ | The $i$th participant. |
| $\mathcal{E}_j$ | The $j$th edge node. |
| $D$ | Total amount of data. |
| $D_i$ | The amount of data for the $i$th participant. |
| $\omega_i$ | Weights of the $i$th participant's machine learning model. |
| $\omega_{glob}$ | Weights of aggregated global machine learning model. |
| $f_i$ | CPU frequency of the $i$th participant. |
| $m_i$ | The number of instructions to process a piece of data in the $i$th participant. |
| $\epsilon_i$ | Local accuracy of the $i$th participant. |
| $T_i^{cmp}$ | Time consumption of local training for the $i$th participant. |
| $r_i$ | Transmission data rate. |
| $B$ | Channel bandwidth. |
| $p_i$ | Transmission power of the $i$th participant. |
| $h_{ij}$ | Channel gain of the link between the $i$th participant and the $j$th edge node. |
| $N_0$ | Gaussian noise. |
| $s_i$ | The number of bits of the $i$th participant's local model parameters. |
| $T_{ij}^{com}$ | Time consumption of communication between the $i$th participant and the $j$th edge node. |
| $a_{ij}$ | Index of participant-edge node pair designation. |
| $q_j$ | Capacity of the $j$th edge node. |
| $\delta$ | Energy threshold of willingness for participation. |
| $c_i$ | Remaining battery percentage of the $i$th participant. |
| $\theta_i$ | Willingness of participation for the $i$th participant. |
| $\mathcal{M}$ | A matching assignment. |
| $\mathcal{A}(\mathcal{E}_j)$ | Acceptable participants set for the $j$th edge node. |
| $\mathcal{A}(\mathcal{N}_i)$ | Acceptable edge nodes set for the $i$th participant. |
| $L(\mathcal{E}_j)$ | Preference list of the $j$th edge node. |
| $L(\mathcal{N}_i)$ | Preference list of the $i$th participant. |

### A. Federated Learning Preliminaries

Due to different customer expectations and daily usage habits, the generated data by each end device can be different, i.e., the data are non-i.i.d. Therefore, the object of federated learning is to cooperatively train a global optimal machine learning model for a certain group of devices or users $\mathcal{N} = \{1, \ldots, N\}$, where the obtained model can be applied to any user within the network. Correspondingly, the individual data set can be denoted as $D_i$, which is in a vector form $(x_i, y_i)$, where $x_i$ describes diverse input data features and $y_i$ represents the output or label. Based on the task requirements, the devices will perform certain local iterations to minimize the loss function $l_i$, i.e.,

$$\min_{\omega} l_i(x_i, \omega; y_i) \tag{1}$$

which is different according to the specific purpose. For instances, the local loss function can be

$$l_i(\omega) = \frac{1}{2}\left(x_i^T \omega - y_i\right), \quad y_i \in \mathbb{R} \tag{2}$$

for a linear regression problem or

$$l_i(\omega) = \max\left\{0, 1 - y_i x_i^T \omega\right\}, \quad y_i \in \{-1, 1\} \tag{3}$$

for a logistic regression problem using the vector support machine. After a certain number of rounds, each device will upload their own model parameters to the aggregator, i.e., the matched MEC server in this work, to perform a weighted average, which can be described as

$$\omega = \frac{\sum_{i=1}^{N} D_i \omega_i}{D} \tag{4}$$

where $D = \sum_{i=1}^{N} D_i$ is the total amount of data. Intuitively, the percentage of local parameters to form the global model is proportional to its data size.

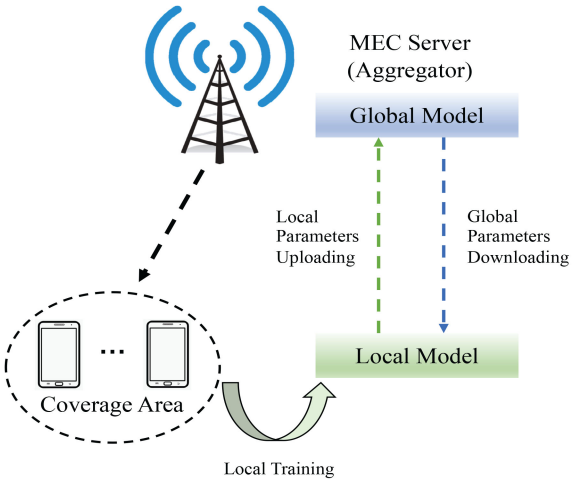



Fig. 1. Federated learning procedures.

Having finished the aggregation process, the calculated parameters will be distributed to all the participated devices and perform local updates. After a certain number of similar interactions, once the maximum number of iterations or required accuracy is achieved, the whole process comes to an end. Overall, the objective function of federated learning can be written as

$$\min_{\omega\in\mathbb{R}^d} J(\omega) = \frac{1}{N}\sum_{i=1}^{N} l_i(\omega). \tag{5}$$

To summarize, each global epoch can be divided into three steps: 1) local computation; 2) participants–aggregator interaction; and 3) recomputation. The corresponding process is illustrated in Fig. 1.

### B. Local Computation Model

Generally, the involved devices can be mobile phones, IoT, and IoV, whose computation abilities are not as powerful as MEC or cloud servers. Hence, their computation tasks are almost accomplished through central processing units (CPUs). We define the CPU frequency for device $\mathcal{N}_i$ as $f_i$. The required number of CPU cycles to process a piece of data sample is $m_i$. We should note that the value $m_i$ will be influenced by the model type, such as the support vector machine (SVM), long-short term memory (LSTM), deep neural network (DNN), convolutional neural network (CNN), and the adopted training methods, such as SGD, minibatch SGD (mBSGD), or Adam. Here, we assume that all the clients are optimizing the same type of machine model via the same kind of training method. Therefore, as suggested in [24], the consumed time for local device $\mathcal{N}_i$ to perform one local iteration can be written as

$$t_i^{\text{cmp}} = \frac{D_i m_i}{f_i}. \tag{6}$$

Obviously, from the perspective of latency, the device with a higher CPU frequency is preferable for the MEC server. Besides, the threshold to upload local parameters to matched server can be defined as achieving a certain local accuracy $\epsilon_i$, where a lower $\epsilon_i$ indicates a higher prediction accuracy. To obtain the desirable accuracy, the requisite number of local iterations can be described as $\log(1/\epsilon_i)$ [25]. Therefore, for device $\mathcal{N}_i$, the consumptive time for one local updates can be calculated as

$$T_i^{\text{cmp}} = \log\left(\frac{1}{\epsilon_i}\right)\frac{D_i m_i}{f_i}. \tag{7}$$

### C. Communication Model

For federated learning, communication happens each time when participated devices upload local parameters and MEC servers broadcast aggregated global parameters. In this work, we adopt time-division medium access (TDMA) technology as the communication protocol. Without loss of generality, for other protocols, similar approaches can be easily extended. Besides, it is assumed that each device is allocated an orthogonal subchannel and the interference brought by neighbor users can be ignored. For device $\mathcal{N}_i$, the transmission rate can be described as

$$r_i = B\log_2\left(1+\frac{p_i h_{ij}}{N_0}\right) \tag{8}$$

where $B$ is the subchannel bandwidth allocated to device $\mathcal{N}_i$, $p_i$ is transmission power, $h_{ij}$ is channel gain between device $\mathcal{N}_i$ and matched MEC server $\mathcal{E}_j$, and $N_0$ is the Gaussian noise. Assume that for all the devices connected to the same MEC server have the same local parameter size $s_i$ bits. Intuitively, the required time for communication can be characterized as

$$T_{ij}^{\text{com}} = \frac{s_i}{B\log_2\left(1+\frac{p_i h_{ij}}{N_0}\right)}. \tag{9}$$

Overall, for device $\mathcal{N}_i$ associated with MEC server $\mathcal{E}_j$ in one single global iteration, the total amount of time consumed can be written as

$$T_{ij} = a_{ij}\left(T_{ij}^{\text{com}} + T_i^{\text{cmp}}\right) \tag{10}$$

where $a_{ij}$ denotes the index for pair designation, while $a_{ij} = 1$ denotes device $\mathcal{N}_i$ is paired with edge node $\mathcal{E}_j$ and *vice versa*.

### D. Problem Formulation

We consider the latency minimization problem for multi-task federated learning in a MEC network with many users $\mathcal{N} = \{1,\ldots,\mathcal{N}_i,\ldots,\mathcal{N}_N\}$ and several edge nodes $\mathcal{E} = \{1,\ldots,\mathcal{E}_j,\ldots,\mathcal{E}_E\}$, where each edge node $\mathcal{E}_j$ is supposed to accomplish the designated machine learning task that is different from all the other edge nodes. Each edge node will be assigned a disparate federated task. The scenario is illustrated in Fig. 2. In this work, we focus on the one global aggregation round delay minimization problem. First, we consider the scenario that the mobile phones, IoT or IoV work as participants and edge server performs as aggregator. Due to the limitation of the edge server coverage and the mobility of IoT and IoV, some devices can only take part in one round of federated learning in practice. Second, actually, the proposed matching can be applied in each global aggregation round, which means if in each round the latency is minimized, then the overall latency is minimized as well. Third, experiments show that if the number of local computation iterations are

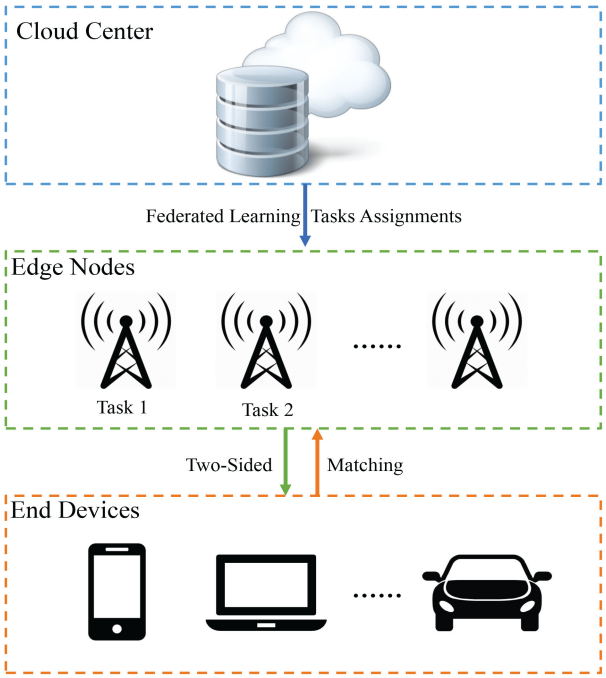



Fig. 2. Multitask federated learning framework in MEC scenario.

sufficient or local accuracy can be achieved sufficiently high, the global model can achieve the high accuracy as well even with one round global communication [26], [27]. Therefore, we only need to consider the latency minimization problem in one communication round.

Due to the fact of diversity of devices computation abilities, from the edge nodes in the latency perspective, the data providers with relatively higher CPU frequency are desirable. On the other hand, the end devices are usually powered by batteries, which are energy constrained. Therefore, in order to proceed tasks as many as possible, providing owned data to the MEC server with a better channel gain $h_{ij}$ is preferable. Besides, when the remained power is lower than a threshold, the device can be unwilling to participate any federated learning. Otherwise, its owned function cannot be supported. From the perspective of a system or network, the aim is to find the optimal device-server pair so as to minimizing the overall time consumption. Because we consider only those device whose battery energy is higher than a certain percentage $\delta$ will take part in federated learning tasks, the parameter $\theta_i$ is introduced, which is denoted as the willingness for participation of the $i$th device. Intuitively, $\theta_i$ is determined by the remained energy percentage, i.e., $\theta_i = 1$ when $\delta \leq c_i$ and *vice versa*. Overall, the corresponding problem can be formulated as follows:

$$\min \sum_{i=1}^{N}\sum_{j=1}^{E}\theta_i T_{ij}$$

s.t.

$$C1: a_{ij} \in \{0,1\} \quad \forall \mathcal{N}_i \in \mathcal{N} \quad \forall \mathcal{E}_j \in \mathcal{E}$$
$$C2: \sum_{j} a_{ij} \leq 1 \quad \forall \mathcal{N}_i \in \mathcal{N}$$
$$C3: \sum_{i} a_{ij} \geq 1 \quad \forall \mathcal{E}_j \in \mathcal{E}$$
$$C4: \sum_{i} a_{ij} \leq q_j \quad \forall \mathcal{E}_j \in \mathcal{E}. \tag{11}$$

Here, $C1$ is the designation element, where $a_{ij} = 1$ denotes device $\mathcal{N}_i$ is connected with edge node $\mathcal{E}_j$; otherwise, they are not paired. $C2$ describes each device can only connects to one edge node. $C3$ represents that for a specific edge node, there will be at least one device connecting to it and assists to accomplish the assigned task. $C4$ denotes the number of connected participants for edge node $\mathcal{E}_j$ cannot exceed its capacity $q_j$.

Evidently, this optimization problem is a 0–1 integer programming problem, which is generally NP-hard to solve. Hence, we are motivated to find a feasible suboptimal solution. Therefore, we utilize a matching theory-based approach: the HR problem with IPL, which will be discussed in detail in the next section.

## IV. Methodology

In the previous section, we have formulated the latency minimization problem as a 0–1 integer programming problem. Because of NP-hardness, we propose a matching theory-based method to find the suboptimal solution. Besides, due to the fact that the number of participants in IoT and IoV scenario can be pretty large, different from the traditional matching game, the preference list cannot be generated completely, which means each side cannot fully obtain all the information of the other side in practice. Therefore, we propose the HR problem with an IPL-based solution in this section. Specifically, the HR modeling is introduced in Section IV-A and the proposed solution is provided in Section IV-B.

### *A. HR Problem Allocation Modeling With Incomplete Preference List*

The HR problem, also sometimes named as the college admission problem, was first proposed by Gale and Shapley [28]. Each year, there will be many medical students looking for hospitals to do practical training. Each hospital will provide a certain number of available positions for them. From the perspective of medical students, each of them will have an order of hospitals, indicating which one is more preferable to join. Simultaneously, hospitals will also review their application materials to decide the order of which ones they prefer to hire. Obviously, in the HR problem, each student can only be assigned to one hospital for practical training. But, for a specific hospital, it will provide a certain number of positions for medical students. Therefore, it is actually a two-sided many-to-one matching problem.

Inspired by the HR problem, we can model the latency minimization problem for multitask federated learning as the HR game. Because in fact, the matching problem between edge nodes and participants is also a two-sided many-to-one problem. The edges nodes can be considered as hospitals that need to hire many participants to finish the federated learning task for them. Meanwhile, the participants can be regarded as medical students, providing their data and computing resources

for edge nodes, and each participant can only be assigned to a specific edge node. Intuitively, the problem involves a set of participants $\mathcal{N}$, a set of edge nodes $\mathcal{E}$, and a set of acceptable pairs $\mathcal{H} = \mathcal{N} \times \mathcal{E}$. The capacity of each edge node $\mathcal{E}_j$ should be a positive integral and every edge node has a set of acceptable participants $\mathcal{A}(\mathcal{E}_j)$, where

$$\mathcal{A}(\mathcal{E}_j) = \{\mathcal{N}_i \in \mathcal{N} : (\mathcal{N}_i, \mathcal{E}_j) \in \mathcal{H}\}. \tag{12}$$

At the same time, each participant $\mathcal{N}_i$ must be accepted by one and only one edge node. Likewise, the acceptable edge nodes options for resident $\mathcal{N}_i$ can be denoted as

$$\mathcal{A}(\mathcal{N}_i) = \{\mathcal{E}_j \in \mathcal{E} : (\mathcal{N}_i, \mathcal{E}_j) \in \mathcal{H}\}. \tag{13}$$

Let us define an agent $k \in \mathcal{N} \times \mathcal{E}$ for the HR problem, which has a preference list in which it ranks $\mathcal{A}(k)$ in a strict order, i.e., no matching candidates share the same preference. Given any participant $\mathcal{N}_i \in \mathcal{N}$ and edge nodes $\mathcal{E}_j, \mathcal{E}_p \in \mathcal{E}$, we can say $\mathcal{N}_i$ prefers $\mathcal{E}_j$ to $\mathcal{E}_p$ if $\mathcal{E}_j$ precedes $p$ on $\mathcal{N}_i$'s preference list, under the condition that $(\mathcal{N}_i, \mathcal{E}_j) \in \mathcal{H}$ and $(\mathcal{N}_i, \mathcal{E}_p) \in \mathcal{H}$. Similarly, the preference relation can be defined for edge nodes.

A matching assignment $\mathcal{M}$ is a subset of $\mathcal{H}$. We can say that $\mathcal{N}_i$ is assigned to $\mathcal{E}_j$ or $\mathcal{E}_j$ is assigned to $\mathcal{N}_i$ if the pair $(\mathcal{N}_i, \mathcal{E}_j) \in \mathcal{M}$. Once $\mathcal{N}_i$ and $\mathcal{E}_j$ are paired and the relation is no longer changeable, the matching $\mathcal{M}$ is stable. The stability notion here implies the robustness to deviations that can be beneficial to both participants and edge nodes [29]. An unstable matching indicates that the participant can change the connected edge node if the altering is beneficial to both of them. However, this kind of unstability is not desirable regarding to the network operation and resource utility. For federated learning, the time of interaction between edge nodes and participants can be more than once. Reconnecting to a new edge node can make lead to the uselessness of current model. Besides, in the multitask federated learning scenario, different edge nodes have different tasks, which means the loss function can be varied. Therefore, for a specific federated learning, all the local training procedures need to be restarted, which is also a waste of time as well as computing resources. Therefore, stability is fatal for matching and here we give the formal stability definition.

*Definition 1:* Let $\mathcal{M}$ be a matching in HR. A pair $(\mathcal{N}_i, \mathcal{E}_j) \in \mathcal{H} \setminus \mathcal{M}$ blocks $\mathcal{M}$, or $(\mathcal{N}_i, \mathcal{E}_j)$ is a blocking pair for $\mathcal{M}$, if the following conditions are satisfied regarding to $\mathcal{M}$.

1) $\mathcal{N}_i$ is unassigned or prefers $\mathcal{E}_j$ to $\mathcal{M}(\mathcal{N}_i)$.
2) $\mathcal{E}_j$ is undersubscribed or prefers $\mathcal{N}_i$ to at least one member of $\mathcal{M}(\mathcal{E}_j)$ (or both).

Then, $\mathcal{M}$ is said to be stable if it admits no blocking pair.

When matching achieves stability, no candidate can find a more preferable partner than the current one and no new matching process will be proposed. In Definition 1, $\mathcal{M}(\mathcal{N}_i)$ indicates the matching of participant $\mathcal{N}_i$ in matching $\mathcal{M}$. In this article, a blocking pair can be defined as a pair of participant and edge node $(i, j)$, where participant $\mathcal{N}_i$ prefers edge node $p$ to its current mate $\mathcal{E}_j$ and edge node $\mathcal{E}_j$ prefers participant $q$ to its current mate $\mathcal{N}_i$. However, traditional HR matching is essentially a two-sided many-to-one matching game. The preference lists of both sides are complete, i.e., for $\forall \mathcal{E}_j \in \mathcal{E}$, each $\mathcal{N}_i \in \mathcal{N}$ has a strict order list to indicate the preference and *vice versa*. However, due to the fact that the number of participants in IoT and IoV scenario can be very large, it is impossible for each edge node to acknowledge the computing abilities or CPU frequencies of all the participants, which leads to the incompleteness of preference list. Therefore, considering the IPL in the HR game is indispensable.

$\mathcal{E}_1 : \mathcal{N}_1\ \mathcal{N}_3\ \mathcal{N}_2\ \mathcal{N}_4\ \mathcal{N}_5$ $\quad$ $\mathcal{N}_1 : \mathcal{E}_2\ \mathcal{E}_1\ \mathcal{E}_3\ \mathcal{E}_4\ \mathcal{E}_5$

$\mathcal{E}_2 : \mathcal{N}_3\ \mathcal{N}_1\ \mathcal{N}_5\ \mathcal{N}_2\ \mathcal{N}_4$ $\quad$ $\mathcal{N}_2 : \mathcal{E}_2\ \mathcal{E}_1\ \mathcal{E}_4\ \mathcal{E}_5\ \mathcal{E}_3$

$\mathcal{E}_3 : \mathcal{N}_2\ \mathcal{N}_1\ \mathcal{N}_5\ \mathcal{N}_4\ \mathcal{N}_3$ $\quad$ $\mathcal{N}_3 : \mathcal{E}_1\ \mathcal{E}_2\ \mathcal{E}_3\ \mathcal{E}_5\ \mathcal{E}_4$

$\mathcal{E}_4 : \mathcal{N}_3\ \mathcal{N}_2\ \mathcal{N}_4\ \mathcal{N}_5\ \mathcal{N}_1$ $\quad$ $\mathcal{N}_4 : \mathcal{E}_3\ \mathcal{E}_1\ \mathcal{E}_4\ \mathcal{E}_2\ \mathcal{E}_5$

$\mathcal{E}_5 : \mathcal{N}_3\ \mathcal{N}_4\ \mathcal{N}_2\ \mathcal{N}_5\ \mathcal{N}_1$ $\quad$ $\mathcal{N}_5 : \mathcal{E}_4\ \mathcal{E}_3\ \mathcal{E}_1\ \mathcal{E}_2\ \mathcal{E}_5$

(a)

$\mathcal{E}_1 : \mathcal{N}_1\ \mathcal{N}_3\ \mathcal{N}_2$ $\quad$ $\mathcal{N}_1 : \mathcal{E}_2\ \mathcal{E}_1\ \mathcal{E}_3\ \mathcal{E}_4\ \mathcal{E}_5$

$\mathcal{E}_2 : \mathcal{N}_3\ \mathcal{N}_1$ $\quad$ $\mathcal{N}_2 : \mathcal{E}_2\ \mathcal{E}_1$

$\mathcal{E}_3 : \mathcal{N}_2\ \mathcal{N}_1$ $\quad$ $\mathcal{N}_3 : \mathcal{E}_1\ \mathcal{E}_2$

$\mathcal{E}_4 : \mathcal{N}_3\ \mathcal{N}_2\ \mathcal{N}_4\ \mathcal{N}_5$ $\quad$ $\mathcal{N}_4 : \mathcal{E}_3\ \mathcal{E}_1\ \mathcal{E}_4$

$\mathcal{E}_5 : \mathcal{N}_3\ \mathcal{N}_4\ \mathcal{N}_2$ $\quad$ $\mathcal{N}_5 : \mathcal{E}_4\ \mathcal{E}_3$

(b)

Fig. 3. Example of matching edge nodes $\{\mathcal{E}_1, \mathcal{E}_2, \mathcal{E}_3, \mathcal{E}_4, \mathcal{E}_5\}$ with end devices $\{\mathcal{N}_1, \mathcal{N}_2, \mathcal{N}_3, \mathcal{N}_4, \mathcal{N}_5\}$ in both complete and IPL cases. The orders indicates each individual's preference list. (a) Matching with CPL. (b) Matching with IPL.

The HR problem with IPL is actually a variant of the standard HR problem. The only difference between them is the completeness of the preference list, which can be described by an example shown in Fig. 3. The majority of notations and definitions in the HR problem can be applied here directly. But for stability, we can redefine as follows.

*Definition 2:* Let $\mathcal{M}$ be a matching in HR with the IPL. A pair $(\mathcal{N}_i, \mathcal{E}_j) \in \mathcal{H} \setminus \mathcal{M}$ blocks $\mathcal{M}$, or $(\mathcal{N}_i, \mathcal{E}_j)$ is a blocking pair for $\mathcal{M}$, if the following conditions are satisfied regarding to $\mathcal{M}$.

1) $\mathcal{N}_i$ is unassigned or prefers $\mathcal{E}_j$ to $\mathcal{M}(\mathcal{N}_i)$.
2) $\mathcal{E}_j$ is unassigned or prefers $\mathcal{N}_i$ to $\mathcal{M}(\mathcal{E}_j)$.

Then, $\mathcal{M}$ is said to be stable if it admits no blocking pair.

Therefore, from the perspective of stability, let us look back to the example in Fig. 3. In Fig. 3(a), for the CPL case, suppose $\mathcal{E}_1$ proposes matching first, so it tries to match with the most desirable candidate, i.e., $\mathcal{N}_1$, according to the preference list. But, $\mathcal{N}_1$ prefers $\mathcal{E}_2$ to $\mathcal{E}_1$ and $\mathcal{E}_2$ is exactly available at this time. So, $(\mathcal{E}_1, \mathcal{N}_1)$ is a blocking pair and $\mathcal{N}_1$ will propose to match with $\mathcal{E}_2$. Likewise, $\mathcal{E}_1$ tries to match with the next candidate $\mathcal{N}_3$. $\mathcal{N}_3$ exactly prefers $\mathcal{E}_1$ best so they will form a pair. Similarly, when all the process is done, we can see the final stable matching results are $(\mathcal{E}_1, \mathcal{N}_3)$, $(\mathcal{E}_2, \mathcal{N}_5)$, $(\mathcal{E}_3, \mathcal{N}_1)$, $(\mathcal{E}_4, \mathcal{N}_2)$, and $(\mathcal{E}_5, \mathcal{N}_4)$. Whereas,

for the matching in Fig. 3(b) with IPL, the doubtless stable pairs are $(\mathcal{E}_1, \mathcal{N}_2)$, $(\mathcal{E}_2, \mathcal{N}_3)$, and $(\mathcal{E}_4, \mathcal{N}_5)$. As for the unpaired individuals, i.e., $\mathcal{E}_3$, $\mathcal{E}_5$, $\mathcal{N}_1$, and $\mathcal{N}_4$, there exist two possibilities to combine them, i.e., $(\mathcal{E}_3, \mathcal{N}_1)$ and $(\mathcal{E}_5, \mathcal{N}_4)$ or $(\mathcal{E}_3, \mathcal{N}_4)$ and $(\mathcal{E}_5, \mathcal{N}_1)$. However, according to Definition 2, we can see that both the options belong to stable matching. Therefore, for the HR problem with IPL, one important conclusion is that the stable matching can be partial. Besides, it has been proved that for HR with IPL problems, there may be more than one stable matching, but their sizes are all the same and one of them can be obtained in poly time [30], [31]. In the next section, the algorithms proposed to solve the corresponding problem will be discussed.

### *B. Participants and Edge Nodes Pairing*

As is discussed above, edge nodes need to hire many participants to finish federated learning task for them. Accordingly, the participants will provide their data and computing resources for edge nodes. Besides, considering the power constraint in practice, edge nodes can only choose those participants whose remaining battery capacity is larger than the predefined threshold $\delta$. Therefore, the acceptable participants set for edge nodes $\mathcal{E}_j$ $(\forall \mathcal{E}_j \in \mathcal{E})$ can be defined as

$$\mathcal{A}(\mathcal{E}_j) = \{\mathcal{N}_i \in \mathcal{N} \mid \delta < c_i\}. \tag{14}$$

For participants, practically they can work for any edge nodes. Therefore, the acceptable edge node set for participants is the universe set of edge nodes, which can be written as

$$\mathcal{A}(\mathcal{N}_i) = \{\mathcal{E}_j \in \mathcal{E}\}. \tag{15}$$

The preference list is based on a private view and can be defined according to utility function. Making use of the computation model described in Section III, i.e., in (7), each edge node is able to calculate the time consumption for each connected participant. Therefore, a preference list of edge node $\mathcal{E}_j$ can be established based on the computation time $T_i^{\text{cmp}*}$, where $T_i^{\text{cmp}*}$ indicates the least time consumption. Correspondingly, we can define the preference list of edge node $\mathcal{E}_j$ as

$$L(\mathcal{E}_j) = T_i^{\text{cmp}*} \quad \forall \mathcal{N}_i \in \mathcal{A}(\mathcal{E}_j). \tag{16}$$

Obviously, $L(\mathcal{E}_j)$ is in an ascending order since less time consumption is always preferable.

When it comes to the participants preference over edge nodes, the power consumption is more important. Because, participants are usually end devices with limited capacity batteries. Apart from taking part in the invited federated learning tasks, they need to consider to maintain enough power and time to perform their normal functionalities as well. Therefore, according to (9), suppose the channel between participant and edge node with a higher channel gain is more desirable. So, the corresponding preference list can be defined as

$$L(\mathcal{N}_i) = T_{ij}^{\text{com}*} \quad \forall \mathcal{E}_j \in \mathcal{A}(\mathcal{N}_i). \tag{17}$$

$L(\mathcal{N}_i)$ is in an ascending order because less communication time is preferable in accordance with a higher channel gain.

**Algorithm 1** Participants and Edge Nodes Pairing With IPL

**Input:** participant set $\mathcal{N}$, edge node set $\mathcal{E}$, remaining battery energy for each participant $c_i$, energy threshold $\delta$, participant CPU frequency $f_i$, channel gain $h_{ij}$, edge node capacity $q_j$;.
All the participants check their remaining battery capacity. Only for those devices meet the requirement $\delta < c_j$ will be the candidates for federated learning tasks.
//Participants build preference list;
**for** i=1:$N$ **do**
  Build the preference list $L(\mathcal{N}_i)$;
**end for**
// Edge nodes build preference list;
**for** j=1:$E$ **do**
  Build the preference list $L(\mathcal{E}_j)$;
**end for**
**while** $\exists \mathcal{E}_j \in \mathcal{E}$ is available and edge node has a non-empty preference list **do**
  //Participants propose to match with edge nodes
  **for** Unmatched $\mathcal{N}_i \in \mathcal{N}$ **do**
    Propose to the top-ranked edge node in participant $\mathcal{N}_i$'s preference list $L(\mathcal{N}_i)$;
    Remove the top-ranked edge node from participant $\mathcal{N}_i$'s preference list $L(\mathcal{N}_i)$;
  **end for**
  //Edge node overflow notification
  **for** $\mathcal{E}_j \in \mathcal{E}$ **do**
    **if** The number of candidates exceeds capacity $q_j$ **then**
      Hold $q_j$ participants based on its preference list $L(\mathcal{E}_j)$ and inform the other participants that they are get rejected from $\mathcal{E}_j$.
    **else**
      Hold all the participants.
    **end if**
  **end for**
**end while**
//Partial matching checking
**for** Unmatched $\mathcal{N}_i \in \mathcal{N}$ **do**
  Randomly assign them to the available edge node.
**end for**
**Output:** Stable many-to-one matching results.

Based on the setting and definitions above, we can apply the many-to-one matching algorithm to find a stable matching solution for participants and edge nodes pairing problem, which is described in Algorithm 1. First, every participant will evaluate its remaining power so as to decide whether it is able to involve a federated learning task. Each participant will generate its preference list based on $L(\mathcal{N}_i)$. If the number of selected participants exceeds edge node capacity, edge node $\mathcal{E}_j$ will only keep those desirable participants within capacity and reject the applications of the others. Then, the unmatched participants will continue to match with the available edge nodes. This process iterates until each participant is either matched or rejected by all the edge nodes based on its preference list, i.e., the matching arrives the stable state. However, since the

preference list is incomplete, the results can be partial matching as illustrated in Fig. 3. Therefore, when the matching arrives at a stable state, checking the assignments for every participant is necessary. If there are still any participants who are willing to join unassigned, they will be assigned to available edge node randomly. For Algorithm 1, we can derive the following proposition.

*Proposition 1:* For Algorithm 1, the proposed many-to-one matching method is able to converge and obtain stable matching results. Besides, as for the performance, the running time of the implementation can achieve $\mathcal{O}(N \times E)$, where $(N \times E)$ denotes all the possible matching pairs.

*Proof:* Let $\mathcal{M}$ be an instance of HR matching. The stable pairs in $\mathcal{M}$ can be found in $\mathcal{O}(N)$ time. The stable matchings in $\mathcal{M}$ can be listed in $\mathcal{O}(E)$ time per matching, after $\mathcal{O}(N)$ preprocessing time. Therefore, the overall running time can achieve $\mathcal{O}(N \times E)$. The corresponding proof can be found in [32] and [33] in details. ■

## V. Numerical Results

In this section, we evaluate our proposed method for matching with the IPL with regard to system latency. Meanwhile, we take matching with the CPL and random matching as baseline methods for a performance comparison. Besides, we change the number of participants, the number of edge nodes, the capacity of each edge node, local accuracy, energy threshold, and incomplete rate to see the performance varieties of the MEC network. In order to control variables, we fix some common parameters for all the participants [9], [10], [12]. The channel bandwidth $B$ is set as 20 MHz. The transmission power for each device $p_i$, the amount of data for each device $D_i$, the number of instructions to process one data sample needed by each participant $m_i$, and the size of local model parameters $s_i$ are assumed the same, which are set as 23 dBm, 1000, 50, and 5 MB, respectively. The Gaussian noise $N_0$ is set as $-96$ dBm. The energy threshold for every participant is assumed the same as well, which is set as 40%. The remaining energy capacity is randomly assigned within $[20, 100]\%$. The CPU frequency is randomly generated within the range $[10, 20]$ MHz. The channel gain of the link between the $i$th participant and the $j$th participant is randomly generated within the interval [10, 20]. The default local accuracy indicator is defined as 0.1, where a lower value of $\epsilon_i$ represents the higher local accuracy. The default percentage of preference missing is set as 10%. For the convenience, the parameters settings are summarized in Table II.

First, we discuss the influence of different participants among participant's average latency. In this case, the number of edge node $E$ is set as 10, where each of them can accept 100 participant at the most. The results are illustrated in Fig. 4.

We increase the number of participants from 1000 to 10 000 by step 1000 to show the change of system latency. Among the three methods shown in Fig. 4, the algorithm with CPL achieves the lowest average latency throughout the variation process, which is understandable. Because with CPL, both sides have entire knowledge to each other, i.e., channel gain and CPU frequency. Therefore, each matching step aims at pairing the higher CPU frequency for edge node and a higher channel gain for the participant, which approaches the lower latency until stable matching is obtained. However, for matching with IPL, due to the incomplete information, the unassigned individuals will be matched randomly with available edge node, which is not desirable and the latency is not the optimal solution. As for the random strategy, it comes by random participants allocation among edge nodes, which gives the highest average latency. Besides, during the variation, the average network latency keeps almost the same for CPL but varies for IPL and random strategies. This is because the random method gives different matching results each time. As for IPL, since we set the missing rate as 10%, the missing preferences introduce uncertainties, which leads to fluctuation.

Then, let us have a look at the influence of different numbers of edge nodes upon network latency. We set the number of users as 5000 and the capacity of edge nodes is 1000. We increase the number of edge nodes from 5 to 10 by step 1. The corresponding results are shown in Fig. 5.

Apparently, CPL achieves the lowest latency because the complete information for preference. The performance of proposed method is in the between of random method and CPL due to a part of matching is assigned randomly instead of the utility functions. Likewise, the random method generates the worst performance reflected by the highest latency.

TABLE II
Table of Key Parameters

| Simulation Parameters | Value |
|---|---|
| Channel bandwidth $B$ | 20MHz |
| Transmission power $p_i$ | 23dBm |
| Data size $D_i$ | 1000 |
| The number of instructions for one piece data processing $m_i$ | 50 |
| The size of local model parameters size $s_i$ | 5MB |
| Gaussian noise $N_0$ | -96dBm |
| Energy threshold $\delta$ | 40% |
| CPU frequency $f_i$ | [10,20]MHz |
| Channel gain $h_{ij}$ | [10,20] |
| Default local accuracy $\epsilon_i$ | 0.1 |

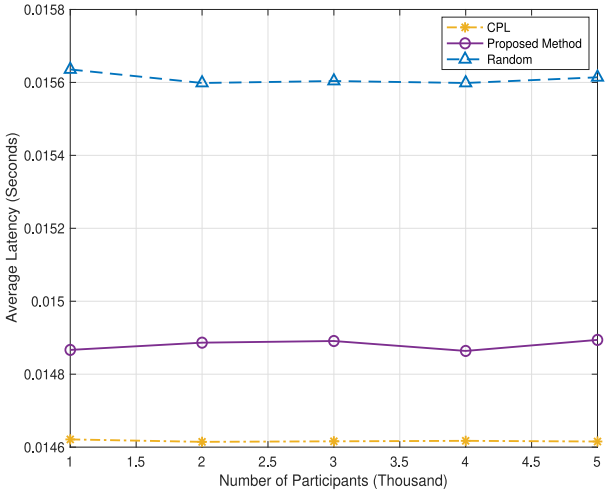



Fig. 4. Average network latency with different number of users.

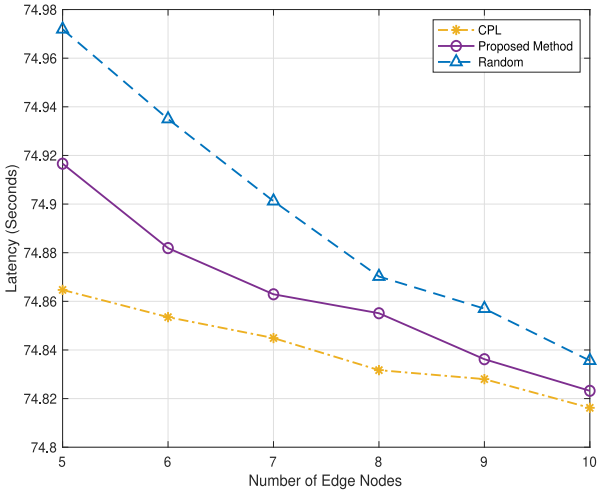



Fig. 5. Network latency with different number of edge nodes.

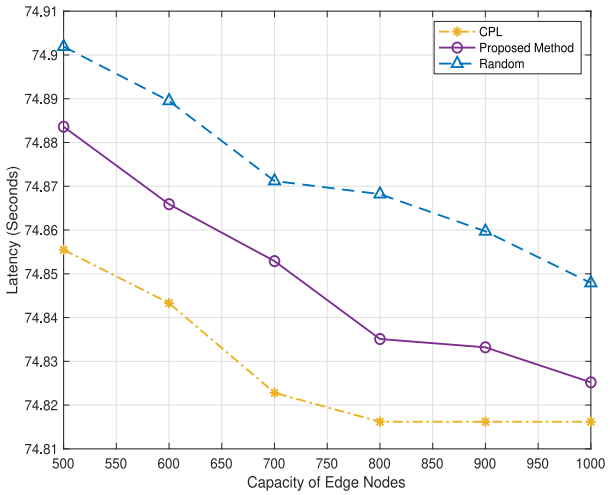



Fig. 6. Network latency with different capacity of edge nodes.

When the edge node number is 5, CPL achieves 0.1072-s less delay than the random method and IPL achieves 0.0533-s less latency than the random method. Besides, we can see that with the increase in the number of edge nodes, the overall latency goes down, which is understandable. When the number of edge nodes is larger, the elements in the acceptable sets $\mathcal{A}(\mathcal{N}_i)$ and $\mathcal{A}(\mathcal{E}_j)$ become larger accordingly, which means the options for participants get augmented. Therefore, each participant is able to find a better assignment with a higher channel gain. Correspondingly, the total number of low latency pairs increases, leading to a relatively lower network delay.

Now, we vary the edge nodes capacity to see the corresponding effect among system latency. We set the number of participants as 5000 and the number of edge nodes as 10. The capacity of each edge node is increased from 500 to 1000 by step 100.

The corresponding results are illustrated in Fig. 6. Overall, the results keep consistent with previous experiments, i.e., random method yields the highest latency, proposed method gives less, and the CPL achieves the lowest latency. Averagely, the latency of random method is 0.0664 s more than the CPL method and 0.0390 s more than the IPL method. Moreover, it is obvious that no matter for which method, the latency reduces with the increase of capacity of edge nodes. Because when the edge node is possessed of a larger capacity, participants can obtain a higher probability to be accepted by the desirable edge node instead of rejection, such that the overall latency can be decreased. Furthermore, for CPL, we can find that the latencies for 800, 900, and 1000 capacity cases are the same. This is because when the edge node capacity is sufficiently large, the matching assignments achieve stability and no more optimal solution can be found. However, for IPL and random methods, due to the embedded uncertainties or randomness, i.e., random preference list missing and random participant-edge node allocation, they cannot achieve the same latency.

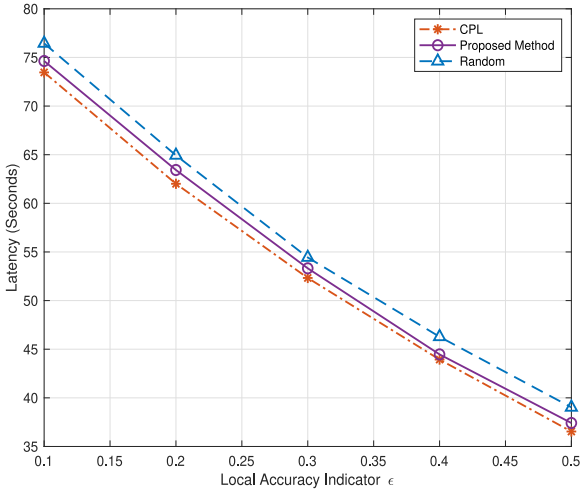



Fig. 7. Network latency with different local accuracy.

Next, we look into the influence of the local accuracy upon network latency. The number of participants, the number of edge nodes, the edge node capacities are set as 5000, 10, and 500, respectively.

The local model accuracy indicator is increased from 0.1 to 0.5 by step 0.1, which represents local accuracy decreases gradually. Corresponding results are illustrated in Fig. 7. Still, the performance of CPL is the best and the random method gives the largest latency. Statistically, the random strategy yields 2.5691 s more delay than CPL and 1.5724 s more latency than IPL. Macroscopically, with a higher accuracy, the latency is much larger as well. Actually, what is affected by local accuracy is the computational time. According to (7), the total number of local computation iterations is in exponential growth with the decay of $\epsilon$. Therefore, in order to achieve a relatively higher local accuracy, the required rounds of computation iterations increase explosively, resulting in a longer time delay.

Also, we change the energy threshold to find the corresponding influence among system latency. The number of participants, the number of edge nodes, the edge node capacity are set as 5000, 10, and 500, respectively. The participation willingness or energy threshold is increased from 40% to 60% by step 5%. The numerical results are shown in Fig. 8.

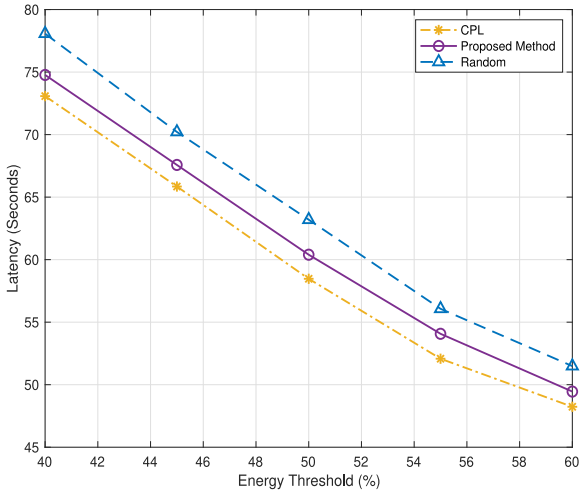



Fig. 8. Network latency with different energy threshold.

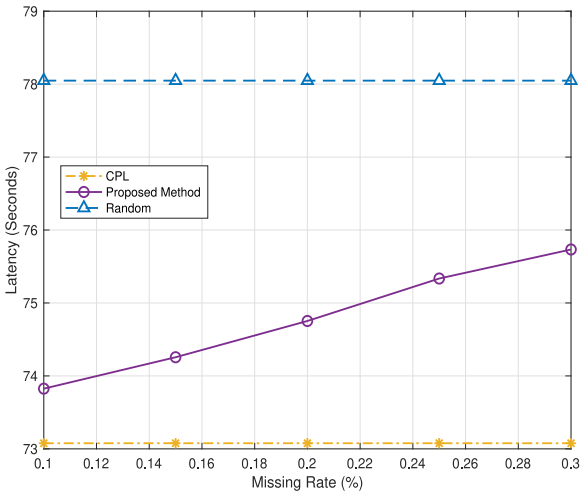



Fig. 9. Network latency with different preference list missing rate.

Apparently, the random method gives the highest network latency while CPL gives the lowest, which is consistent with previous experiments. On average, the random method generates 4.2652 s higher latency than CPL and 2.5547 s higher latency than IPL. In general, with the increase in energy threshold $\delta$, the total network latency decreases accordingly. The index of willingness for participation is determined by remained energy percentage. Only when $\delta$ is less than $c_i$, $\theta_i$ equals 1, i.e., $\mathcal{N}_i$ will join the federated learning task. Hence, as energy threshold increases, the total number of participants involving those in the learning process will decrease. Therewith, the overall network latency can be reduced correspondingly.

Finally, we adjust the parameter of the preference list missing rate, which is from 10% to 30% by step 5%. The number of participants, the number of edge nodes, the edge node capacity are set as 5000, 10, and 500, respectively. The corresponding results are illustrated in Fig. 9.

The overall trend is that with the increase of missing rate, the network latency becomes larger. We can find that the performance approaches to CPL with small missing rate but similar to random method with a larger missing rate. This is because if the missing rate is higher, the probability of partial matching increases as well. Correspondingly, the number of unassigned individuals will be aggrandized. Since unassigned individuals will be matched with available edge node randomly, which is not based on utility function. Consequently, the obtained latency is not the optimal solution. Therefore, the performance get closer to random strategy. In an extreme case, when the missing rate is 100%, IPL will be exactly the same as random allocation. On the contrary, when the missing rate is small, which means the built preference list is relatively complete, the system latency can achieve similar value compared with CPL. When the missing rate is sufficiently small, i.e., 0%, IPL will turn into CPL case.

## VI. Conclusion

In this work, we studied the low-latency problem for multi-task federated learning in the MEC networks. Considering in the large-scale IoT scenario, it is not possible for edge nodes and end devices to obtain the complete information of the other side, which means building the CPL is impractical. Therefore, we proposed a method to deal with the two-sided many-to-one matching with the IPL. The simulation results showed that the performance of our proposed method is close to the performance of CPL, although there is a small gap between them due to information missing. Besides, we also discussed the influence of number of participants, the number of edge nodes, the edge node capacity, local accuracy, energy threshold, and preference list missing rate among network latency. Evidently, the network latency is positively related to the missing rate while is negatively correlated with number of edge nodes, capacity of edge nodes, energy threshold, and local accuracy indicator.

## References

[1] D. Reinsel, J. Gantz, and J. Rydning, "The digitization of the world from edge to core," IDC, Framingham, MA, USA, White Paper, 2018.
[2] J. Ren, G. Yu, and G. Ding, "Accelerating DNN training in wireless federated edge learning system," 2019. [Online]. Available: arXiv:1905.09712
[3] Y. Zhang and Q. Yang, "A survey on multi-task learning," 2018. [Online]. Available: arXiv:1707.08114
[4] M. Hao, H. Li, X. Luo, G. Xu, H. Yang, and S. Liu, "Efficient and privacy-enhanced federated learning for industrial artificial intelligence," *IEEE Trans. Ind. Informat.*, vol. 16, no. 10, pp. 6532–6542, Oct. 2020.
[5] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated learning: Strategies for improving communication efficiency," 2016. [Online]. Available: arXiv:1610.05492
[6] D. Chen, Y.-C. Liu, B. Kim, J. Xie, C. S. Hong, and Z. Han, "Edge computing resources reservation in vehicular networks: A meta-learning approach," *IEEE Trans. Veh. Technol.*, vol. 69, no. 5, pp. 5634–5646, May 2020.
[7] Q. Yang, Y. Liu, T. Chen, and Y. Tong, "Federated machine learning: Concept and applications," *ACM Trans. Intell. Syst. Technol. (TIST)*, vol. 10, no. 2, pp. 1–19, Jan. 2019.
[8] K. Bonawitz *et al.*, "Practical secure aggregation for privacy-preserving machine learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, Dallas, TX, USA, Oct. 2017, pp. 1175–1191.
[9] Y. Zhao, M. Li, L. Lai, N. Suda, D. Civin, and V. Chandra, "Federated learning with non-IID data," 2018. [Online]. Available: arXiv:1806.00582.
[10] S. Wang *et al.*, "Adaptive federated learning in resource constrained edge computing systems," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1205–1221, Jun. 2019.

[11] F. Sattler, S. Wiedemann, K.-R. Müller, and W. Samek, "Robust and communication-efficient federated learning from non-i.i.d. data," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 9, pp. 3400–3413, Sep. 2020.
[12] K. Yang, T. Jiang, Y. Shi, and Z. Ding, "Federated learning via over-the-air computation," *IEEE Trans. Wireless Commun.*, vol. 19, no. 3, pp. 2022–2035, Mar. 2020.
[13] J. Kang, Z. Xiong, D. Niyato, H. Yu, Y.-C. Liang, and D. I. Kim, "Incentive design for efficient federated learning in mobile networks: A contract theory approach," in *Proc. IEEE VTS Asia–Pac. Wireless Commun. Symp. (APWCS)*, Signapore, Aug. 2019, pp. 1–5.
[14] C. Dinh *et al.*, "Federated learning over wireless networks: Convergence analysis and resource allocation," Mar. 2020. [Online]. Available: arXiv:1910.13067
[15] W. Y. B. Lim *et al.*, "Towards federated learning in UAV-enabled Internet of Vehicles: A multi-dimensional contract-matching approach," Apr. 2020. [Online]. Available: arXiv:2004.03877
[16] H. Zhu and Y. Jin, "Multi-objective evolutionary federated learning," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 4, pp. 1310–1322, Apr. 2020.
[17] D. Manlove, *Algorithmics of Matching Under Preferences*, vol. 2. Hackensack, NJ, USA: World Sci., 2013.
[18] C. Su, F. Ye, L.-C. Wang, L. Wang, Y. Tian, and Z. Han, "UAV-assisted wireless charging for energy-constrained IoT devices using dynamic matching," *IEEE Internet Things J.*, vol. 7, no. 6, pp. 4789–4800, Jun. 2020.
[19] Y. Gu, Z. Chang, M. Pan, L. Song, and Z. Han, "Joint radio and computational resource allocation in IoT fog computing," *IEEE Trans. Veh. Technol.*, vol. 67, no. 8, pp. 7475–7484, Aug. 2018.
[20] N. Sharghivand, F. Derakhshan, L. Mashayekhy, and L. M. Khanli, "An edge computing matching framework with guaranteed quality of service," *IEEE Trans. Cloud Comput.*, early access, Jun. 29, 2020, doi: 10.1109/TCC.2020.3005539.
[21] S. Seng, C. Luo, X. Li, H. Zhang, and H. Ji, "User matching on blockchain for computation offloading in ultra-dense wireless networks," *IEEE Trans. Netw. Sci. Eng.*, early access, Jun. 9, 2020, doi: 10.1109/TNSE.2020.3001081.
[22] R. Fantacci and B. Picano, "A matching game with discard policy for virtual machines placement in hybrid cloud-edge architecture for industrial IoT systems," *IEEE Trans. Ind. Informat.*, vol. 16, no. 11, pp. 7046–7055, Nov. 2020.
[23] X. Huang, R. Yu, S. Xie, and Y. Zhang, "Task-container matching game for computation offloading in vehicular edge computing and networks," *IEEE Trans. Intell. Transp. Syst.*, early access, May 8, 2020, doi: 10.1109/TITS.2020.2990462.
[24] L. U. Khan, M. Alsenwi, Z. Han, and C. S. Hong, "Self organizing federated learning over wireless networks: A socially aware clustering approach," in *Proc. Int. Conf. Inf. Netw. (ICOIN)*, Barcelona, Spain, Jan. 2020, pp. 453–458
[25] N. H. Tran, W. Bao, A. Zomaya, M. N. H. Nguyen, and C. S. Hong, "Federated learning over wireless networks: Optimization model design and analysis," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Paris, France, Apr. 2019, pp. 1387–1395.
[26] Y. Chen, X. Sun, and Y. Jin, "Communication-efficient federated deep learning with layerwise asynchronous model update and temporally weighted aggregation," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 10, pp. 4229–4238, Oct. 2020.
[27] D. Chen *et al.*, "Federated learning based mobile edge computing for augmented reality applications," in *Proc. Int. Conf. Comput. Netw. Commun. (ICNC)*, Big Island, HI, USA, Feb. 2020, pp. 767–773.
[28] D. Gale and L. S. Shapley, "College admissions and the stability of marriage," *Amer. Math. Monthly*, vol. 120, no. 5, pp. 386–391, 2013.
[29] N. Raveendran *et al.*, "Cyclic three-sided matching game inspired wireless network virtualization," *IEEE Trans. Mobile Comput.*, vol. 20, no. 2, pp. 416–428, Feb. 2021.
[30] D. Gale and M. Sotomayor, "Ms. machiavelli and the stable matching problem," *Amer. Math. Monthly*, vol. 92, no. 4, pp. 261–268, 1985.
[31] D. Gale and M. Sotomayor, "Some remarks on the stable matching problem," *Discrete Appl. Math.*, vol. 11, no. 3, pp. 223–232, Jul. 1985.
[32] I. P. Gent and P. Prosser, "An empirical study of the stable marriage problem with ties and incomplete lists," in *Proc. 15th Eur. Conf. Artif. Intell.*, Amsterdam, The Netherlands, Aug. 2002, pp. 141–145.
[33] K. Iwama and S. Miyazaki, "A survey of the stable marriage problem and its variants," in *Proc. Int. Conf. Informat. Educ. Res. Knowl. Circulating Soc. (ICKS)*, Tokyo, Japan, Mar. 2008, pp. 131–136.

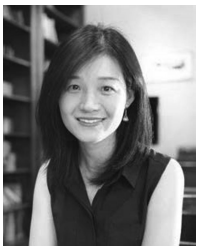
**Dawei Chen** (Student Member, IEEE) received the B.S. degree in telecommunication engineering from the Huazhong University of Science and Technology, Wuhan, China, in 2015. He is currently pursuing the Ph.D. degree with the Department of Electrical and Computer Engineering, University of Houston, Houston, TX, USA.

His research interests include deep learning, federated learning/analytics, cloud/edge computing, and wireless networks.

**Choong Seon Hong** (Senior Member, IEEE) received the B.S. and M.S. degrees in electronic engineering from Kyung Hee University, Seoul, South Korea, in 1983 and 1985, respectively, and the Ph.D. degree from Keio University, Tokyo, Japan, in 1997.

In 1988, he joined KT, Seoul, where he was involved in broadband networks as a member of the Technical Staff. Since 1993, he has been with Keio University. He was with the Telecommunications Network Laboratory, KT, as a Senior Member of Technical Staff and as the Director of the Networking Research Team until 1999. Since 1999, he has been a Professor with the Department of Computer Science and Engineering, Kyung Hee University. His research interests include future Internet, intelligent edge computing, network management, and network security.

Prof. Hong has served as the General Chair, the TPC Chair/Member, or an Organizing Committee Member of international conferences, such as the Network Operations and Management Symposium, International Symposium on Integrated Network Management, Asia–Pacific Network Operations and Management Symposium, End-to-End Monitoring Techniques and Services, IEEE Consumer Communications and Networking Conference, Assurance in Distributed Systems and Networks, International Conference on Parallel Processing, Data Integration and Mining, World Conference on Information Security Applications, Broadband Convergence Network, Telecommunication Information Networking Architecture, International Symposium on Applications and the Internet, and International Conference on Information Networking. He was an Associate Editor of the IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT and the IEEE JOURNAL OF COMMUNICATIONS AND NETWORKS and the *International Journal of Network Management* and an Associate Technical Editor of the *IEEE Communications Magazine*. He currently serves as an Associate Editor for the *International Journal of Network Management* and *Future Internet Journal*. He is a member of the Association for Computing Machinery, the Institute of Electronics, Information and Communication Engineers, the Information Processing Society of Japan, the Korean Institute of Information Scientists and Engineers, the Korean Institute of Communications and Information Sciences, the Korean Information Processing Society, and the Open Standards and ICT Association.



**Li Wang** (Senior Member, IEEE) received the Ph.D. degree from Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 2009.

She is currently a Full Professor with the School of Computer Science, National Pilot Software Engineering School, BUPT, where she is also an Associate Dean and the Head of the High Performance Computing and Networking Laboratory. She is also a Member of the Key Laboratory of the Universal Wireless Communications, Ministry of Education, Beijing, China. She also held visiting positions with the School of Electrical and Computer Engineering, Georgia Tech, Atlanta, GA, USA, from December 2013 to January 2015, and the Department of Signals and Systems, Chalmers University of Technology, Gothenburg, Sweden, from August 2015 to November 2015 and July 2018 to August 2018. She has authored/coauthored almost 50 journal papers and two books. Her current research interests include wireless communications, distributed networking and storage, vehicular communications, social networks, and edge AI.

Prof. Wang was a recipient of the 2013 Beijing Young Elite Faculty for Higher Education Award, the Best Paper Awards from several IEEE conferences, e.g., IEEE ICCC 2017, IEEE GLOBECOM 2018, IEEE WCSP 2019, and so forth. She was also a recipient of the Beijing Technology Rising Star Award in 2018. She was the Symposium Chair of IEEE ICC 2019 on Cognitive Radio and Networks Symposium and a Tutorial Chair of IEEE VTC-Fall 2019. She also serves as the Vice Chair of Meetings and Conference Committee for IEEE Communication Society Asia Pacific Board for the term of 2020–2021, and chairs the special interest group on Social Behavior Driven Cognitive Radio Networks for IEEE Technical Committee on Cognitive Networks. She has served on TPC of multiple IEEE conferences, including IEEE Infocom, Globecom, International Conference on Communications, IEEE Wireless Communications and Networking Conference, and IEEE Vehicular Technology Conference in recent years. She currently serves on the editorial boards for IEEE TRANSACTIONS ON VEHICULAR TECHNOLOGY, IEEE TRANSACTIONS ON GREEN COMMUNICATIONS AND NETWORKING, *Computer Networks*, IEEE ACCESS, and *China Communications*.

**Yiyong Zha** received the B.S. degree in applied physics from the University of Science and Technology of China, Hefei, China, in 2008, and the Ph.D. degree in electrical and computer engineering from the University of Houston, Houston, TX, USA, in 2013.

He worked as a Researcher in networking and cloud industry after graduation. He has many published papers and patents. He currently serves as an Expert Engineer with Tencent Technology, Shenzhen, China. His research interest is mainly focused on 5G, edge computing, and real-time service on cloud.

**Yunfei Zhang** received the B.S. degree from Northern Jiaotong University, Beijing, China, in 1999, and the Ph.D. degree from Beijing Jiaotong University, Beijing, China, in 2005.

He is the Director of Tencent Future Network Laboratory, Shenzhen, China. He is also an Adjunct Professor with Beijing Jiaotong University, Beijing, China. He has published more than 40 papers in international journals and conferences and more than 200 patents in the fields of wireless network, cloud computing, distributed systems, streaming media, Internet of Things, and mobile phone. He has rich experience in Internet and wireless standards, established and chaired the working group of P2P Streaming Media Protocol in IETF, served as an ITU-T Editor, led the first 3GPP Standard Team in Chinese smart phone vendors, and issued more than ten international standards in IETF and ITU-T.

Dr. Zhang serves as an IEEE GLOBECOM 2019 Keynote Speech Session and ICC 2019 Forum Co-Chair.

**Xin Liu** received the B.S. degree from Northwestern Polytechnical University, Xi'an, China, in 1994.

He has more 20 years of R&D experience Internet and communication industry. Starting from 3G to 5G, he actively leading the Technology Platform RD Team, Tencent Technology Company Ltd., Shenzhen, China. He has many publications and patents on mobile Internet and communication industry. He is currently the Department Manager of Platform & Content Group, Tencent. He is leading the Future Network Team that is contributing to standard bodies, such as 3GPP, ITU, and IETF. Also, he serves as the Director of the Linux Foundation and committing to opensource collaboration in the industry.

**Zhu Han** (Fellow, IEEE) received the B.S. degree in electronic engineering from Tsinghua University, Beijing, China, in 1997, and the M.S. and Ph.D. degrees in electrical and computer engineering from the University of Maryland, College Park, MD, USA, in 1999 and 2003, respectively.

From 2000 to 2002, he was a Research and Development Engineer with JDS Uniphase Corporation, Germantown, MD, USA. From 2003 to 2006, he was a Research Associate with the University of Maryland. From 2006 to 2008, he was an Assistant Professor with Boise State University, Boise, ID, USA. He is currently a John and Rebecca Moores Professor with the Electrical and Computer Engineering Department and the Computer Science Department, University of Houston, Houston, TX, USA. His research interests include wireless resource allocation and management, wireless communications and networking, game theory, big data analysis, security, and smart grid.

Dr. Han received the NSF Career Award in 2010, the Fred W. Ellersick Prize of the IEEE Communication Society in 2011, the EURASIP Best Paper Award for the *Journal on Advances in Signal Processing* in 2015, the IEEE Leonard G. Abraham Prize in the field of Communications Systems (Best Paper Award in IEEE JOURNAL ON SELECTED AREAS IN COMMUNICATIONS) in 2016, and several best paper awards in IEEE conferences. He is also the Winner of the 2021 IEEE Kiyo Tomiyasu Award, for outstanding early to mid-career contributions to technologies holding the promise of innovative applications, with the following citation: "for contributions to game theory and distributed management of autonomous communication networks." He has been 1% Highly Cited Researcher since 2017 according to Web of Science. He was an IEEE Communications Society Distinguished Lecturer from 2015 to 2018. He has been an AAAS Fellow since 2019 and an ACM Distinguished Member since 2019.