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 multi-task federated learning in a multi-access 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 (IoTs) 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 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 complete preference list case.

Index Terms—multi-access edge computing, multi-task federated learning, matching theory, incomplete preference list.

I. INTRODUCTION

THE last decade has witnessed an unprecedented improvement and prosperity of machine learning techniques and applications, such as face recognition, driverless vehicles, and autonomous disease diagnose, etc. On the one hand, such rapid development is heavily dependent on the tremendous growth of multi-access edge computing (MEC) networks, there will be many available edge nodes deployed at the edge, 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 firstly proposed by [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. Firstly, 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, stochastic gradient descent 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 pre-defined accuracy is achieved.

Intuitively, there are several reasons to adopt federated learning in practical situation. Firstly, with the development of multi-access 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

Matching Theory Based Low-Latency Scheme for Multi-Task 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
computing resources, edge node can be sufficiently powerful to process federated learning tasks instead of using central-ized data centers so as to reduce the transmission latency [6]. Secondly, 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 extent such that the privacy can be guaranteed [8]. Thirdly, 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 non-identical and independent distribution (non-i.i.d.) data [9], which is suitable for large scale IoTs scenarios.

With all the alluring benefits above, federated learning is also faced with new challenges to tackle. On the 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 that 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. Secondly, 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 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. Thirdly, there are many available edge nodes in a MEC network, while each node can work as an aggregator actually. Therefore, how to parallelly perform multiple federated learning tasks, i.e., multi-task 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 based method towards a low-latency purpose for multi-task federated learning in the MEC network. The contribution of this work can be summarized as follows:

- Firstly, we analyze and formulate the low-latency problem for multi-task federated learning in the MEC network from computation and communication perspectives. Then, the corresponding formulation is given to consider multi-task federated learning. Most existing literature only considers one single federated learning problem.
- Secondly, we model the low-latency problem as the hospitals-residents (HR) matching problem. Besides, considering that the number of participants in IoTs 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 is not practical. Therefore, we propose a matching with the incomplete preference list method to solve the problem so as to get closer to reality, which is seldom done by existing literature.
- Thirdly, 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 complete preference list case with small gap between them due to information missing.

The rest of this paper 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 multi-task federated learning in the MEC network. In Section IV, related matching definitions are given and the proposed matching algorithm with the incomplete preference list 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. [10] analyzes the convergence bound for federated learning with non-i.i.d. data and proposes a control algorithm to achieve an optimal trade-off between local updates and global aggregation considering a resource budget constrain. [11] proposes a sparse ternary compression framework to reduce the communication cost for federated learning with non-i.i.d. data. [12] proposes 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. [13] proposes 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. [14] proposes a fast convergence algorithm to find an optimal trade-off 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. [15] proposes a multi-dimensional contract-matching incentive framework to maximization the profit of model owners in a unmanned aerial vehicle (UAV) enabled (Internet of Vehicles) IoVs scenario. [16] utilizes a multi-objective evolutionary algorithm to simultaneously minimize the communication cost and maximize the global model accuracy. However, regarding these work, [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 multi-task federated learning is not considered.

As for 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].
[18] proposes an algorithm that combines the Markov decision process with random serial dictatorship matching to solve the UAV-assisted charging problem for energy constrained devices. [19] proposes a student project allocation game based matching method to address the joint radio and computation resource allocation problem for IoTs in the fog computing scenario. [20] proposes a two-sided matching solution for IoTs and edge nodes matching problem to reduce the average service time so that quality of service (QoS) requirements can be achieved. [21] develops an efficient task-virtual machine matching algorithm that jointly considers task execution time and energy consumption to make computation offloading decisions in ultra-dense wireless networks. [22] proposes a matching based strategy for virtual machine placement so as to minimize the system response time and requests dropping for industrial IoTs applications. [23] applies 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 work mentioned above, the matching game is considers under the complete preference list situation, which is not practical in large-scale IoTs applications.

III. SYSTEM MODEL AND PROBLEM FORMULATION

In this section, we introduce the preliminaries for federated learning in Subsection III-A. Then, the computation and communication model are discussed in Subsections III-B and III-C, respectively. At the last, we describe the scenario in details and provide the corresponding formulation for multi-task federated learning latency minimization problem within the MEC network in Subsection III-D. For a clear understanding of parameters and symbols in this paper, their definitions and descriptions are provided in Table I in details.

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 \( N = \{1, \ldots, N\} \), where the obtained model can be applied to any user within the network. Correspondingly, the individual dataset can be denoted as \( D_i \), which is in a vector form \((x_i, y_i)\), where \(x_i\) describes a diverse input data features and \(y_i\) represents the output or label. Based on the task requirements, the devices will perform a certain local iterations to minimize the loss function \(l_i\), i.e.,

\[
\min_\omega l_i(x_i, \omega; y_i),
\]

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

\[
l_i(\omega) = \frac{1}{2} (x_i^T \omega - y_i)^2, \quad y_i \in \mathbb{R}
\]

for linear regression problem or

\[
l_i(\omega) = \max \{0, 1 - y_i x_i^T \omega\}, \quad y_i \in \{-1, 1\}
\]

for a logistic regression problem using 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

\[
\omega = \frac{\sum_{i=1}^{N} D_i \omega_i}{\sum_{i=1}^{N} D_i},
\]

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.

Having finished 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 iteration or required accuracy is achieved, the whole process comes to an end. Overall, the objective function of federated learning can be written as

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

To summarize, each global epoch can be divided into three steps: local computation, participants-aggregator interaction, and recomputation. The corresponding process is illustrated in Fig. 1.
This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/JIOT.2021.3053283, IEEE Internet of Things Journal

B. Local Computation Model

Generally, the involved devices can be mobile phones, IoTs, and IoVs, 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 $N_i$ 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 support vector machine (SVM), long-term short memory (LSTM), deep neural network (DNN), convolutional neural network (CNN), and the adopted training methods, such as stochastic gradient descent (SGD), mini-batch stochastic gradient descent (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 is suggested in [24], the consumed time for local device $N_i$ to perform one local iteration can be written as

$$
\epsilon_i^{cmp} = \frac{D_i m_i}{f_i}.
$$

(6)

Obviously, from the perspective of latency, the device with a higher CPU frequency is preferable for 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 $\log(\frac{1}{\epsilon_i})$ [25]. Therefore, for device $N_i$, the consumptive time for one local updates can be calculated as

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

(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.

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 sub-channel and the interference brought by neighbor users can be ignored. For device $N_i$, the transmission rate can be described as

$$
\frac{r_i}{B} = \log\left(1 + \frac{p_i h_{ij}}{N_0}\right),
$$

(8)

where $B$ is the sub-channel bandwidth allocated to device $N_i$, $p_i$ is transmission power, $h_{ij}$ is channel gain between device $N_i$ and matched MEC server $E_j$, and $N_0$ is the Gaussian noise. Suppose 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

$$
T_i^{com} = \frac{s_i}{B \log\left(1 + \frac{N_0 h_{ij}}{p_i}\right)}.
$$

(9)

Overall, for device $N_i$ associated with MEC server $E_j$ in one single global iteration, the total time of consumed can be written as

$$
T_{ij} = a_{ij}(T_{ij}^{com} + T_{ij}^{cmp}),
$$

(10)

where $a_{ij}$ denotes the index for pair designation, while $a_{ij} = 1$ denotes device $N_i$ is paired with edge node $E_j$, and vice versa.

D. Problem Formulation

We consider the latency minimization problem for multi-task federated learning in a MEC network with many users $N = \{1, ..., N_i, ..., N_N\}$ and several edge nodes $E = \{1, ..., E_j, ..., E_E\}$, where each edge node $E_j$ is supposed to accomplish the designated machine learning task which is different from all the other edge nodes. And 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. Firstly, we consider the scenario that the mobile phones, IoTs or IoVs work as participants and edge server performs as aggregator. Due to the limitation of edge server coverage and the mobility of IoTs and IoVs, some devices can only take part in one round of federated learning in practice. Secondly, 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. Thirdly, experiments show that if the number of local computation iterations are 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]. Hence, we are motivated to find a feasible suboptimal solution. Therefore, we utilize a matching theory based approach: the HR problem with incomplete preference list, which will be discussed in details 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 IoTs and IoVs 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 HR problem with incomplete preference list based solution in this section. Specifically, the HR modelling is introduced in Subsection IV-A, and the proposed solution is provided in Subsection IV-B.

A. HR Problem Allocation Modelling with Incomplete Preference List

The HR problem, also sometimes named as the college admission problem was firstly proposed by Gale and Shapley [28]. Each year, there will be many medical students looking for hospitals to do practical training. And 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 multi-task 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 which need to hire many participants to finish 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 \( N \), a set of edge nodes \( E \), and a set of acceptable pairs \( H = N \times E \). The capacity of each edge node \( E_j \) should be a positive integral and every edge node has a set of acceptable participants \( A(E_j) \), where

\[
A(E_j) = \{N'_i \in N : (N'_i, E_j) \in H\}. \tag{12}
\]

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

\[
A(N'_i) = \{E_j \in E : (N'_i, E_j) \in H\}. \tag{13}
\]
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 \( A(k) \) in a strict order, i.e., no matching candidates share the same preference. Given any participant \( N_i \in \mathcal{N} \) and edge nodes \( E_j, E_p \in \mathcal{E} \), we can say \( N_i \) prefers \( E_j \) to \( E_p \) if \( E_j \) precedes \( p \) on \( N_i \)'s preference list, under the condition that \( (N_i, E_j) \in \mathcal{H} \) and \( (N_i, E_p) \in \mathcal{H} \). Similarly, the preference relation can be defined for edge nodes.

A matching assignment \( \mathcal{M} \) is a subset of \( \mathcal{H} \). We can say \( N_i \) is assigned to \( E_j \) or \( E_j \) is assigned to \( N_i \) if the pair \( (N_i, E_j) \in \mathcal{M} \). Once \( N_i \) and \( E_j \) is 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 participant can change the connected edge node if the altering is beneficial to both of them. However, this kind of instability is not desirable regarding to the network operation and resources 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 multi-task federated learning scenario, different edge nodes have different tasks, which means the loss function can be various. 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 \( (N_i, E_j) \in \mathcal{H} \setminus \mathcal{M} \) blocks \( \mathcal{M} \), or \( (N_i, E_j) \) is a blocking pair for \( \mathcal{M} \), if the following conditions are satisfied regarding to \( \mathcal{M} \):

1) \( N_i \) is unassigned or prefers \( E_j \) to \( \mathcal{M}(N_i) \);  
2) \( E_j \) is under-subscribed or prefers \( N_i \) to at least one member of \( \mathcal{M}(E_j) \) (or both).

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

When matching achieves stable, no candidate can find a more preferable partner than current one and no new matching process will be proposed. In Definition 1, \( \mathcal{M}(N_i) \) indicates the matching of participant \( N_i \) in matching \( \mathcal{M} \). In this paper, a blocking pair can be defined as a pair of participant and edge node \((i,j)\), where participant \( N_i \) prefers edge node \( p \) to its current mate \( E_j \) and edge node \( E_j \) prefers participant \( q \) to its current mate \( N_i \). However, traditional HR matching is essentially a two-sided many-to-one matching game. And the preference list of both sides are complete, i.e., for \( \forall E_j \in \mathcal{E} \), each \( N_i \in \mathcal{N} \) has a strict order list to indicate the preference, and vice versa. However, due to the fact that the number of participants in IoTs and IoVs 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 incomplete preference list in HR game is indispensable.

HR problem with incomplete preference list is actually a variant of standard HR problem. The only difference between them is the completeness of preference list, which can be described by an example shown in Fig. 3. The majority of participants prefer complete preference list, so we define a matching that is stable under complete preference list as follows:

- \( E_1, E_2, E_3, E_4, E_5 \):
  - \( N_1, N_2, N_3, N_4, N_5 \):
    - \( N_1 \) prefers \( E_1 \) to \( E_2 \) to \( E_3 \) to \( E_4 \) to \( E_5 \) and \( N_2 \) prefers \( E_2 \) to \( E_1 \) to \( E_3 \) to \( E_4 \) to \( E_5 \) and \( N_3 \) prefers \( E_3 \) to \( E_1 \) to \( E_2 \) to \( E_4 \) to \( E_5 \) and \( N_4 \) prefers \( E_4 \) to \( E_1 \) to \( E_2 \) to \( E_3 \) to \( E_5 \) and \( N_5 \) prefers \( E_5 \) to \( E_1 \) to \( E_2 \) to \( E_3 \) to \( E_4 \).

(a) Matching with Complete Preference List.

(b) Matching with Incomplete Preference List.

Fig. 3. The example of matching edge nodes \( \{E_1, E_2, E_3, E_4, E_5\} \) with end devices \( \{N_1, N_2, N_3, N_4, N_5\} \) in both complete and incomplete preference list cases. The orders indicates each individual’s preference list.

Note that notations and definitions in the HR problem can be applied here directly. But for stability, we can redefine as the following.

**Definition 2**: Let \( \mathcal{M} \) be a matching in HR with the incomplete preference list. A pair \( (N_i, E_j) \in \mathcal{H} \setminus \mathcal{M} \) blocks \( \mathcal{M} \), or \( (N_i, E_j) \) is a blocking pair for \( \mathcal{M} \), if the following conditions are satisfied regarding to \( \mathcal{M} \):

1) \( N_i \) is unassigned or prefers \( E_j \) to \( \mathcal{M}(N_i) \);  
2) \( E_j \) is unassigned or prefers \( N_i \) to at least one member of \( \mathcal{M}(E_j) \) (or both).

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 complete preference list case, suppose \( E_1 \) proposes matching firstly, so it tries to match with the most desirable candidate, i.e., \( N_1 \), according to preference list. But \( N_1 \) prefers \( E_2 \) to \( E_1 \) and \( E_2 \) is exactly available at this time. So \( (E_1, N_1) \) is a blocking pair and \( N_1 \) will propose to match with \( E_2 \). Likewise, \( E_1 \) tries to match with the next candidate \( N_2 \). \( N_3 \) exactly prefers \( E_1 \) best so they will form a pair. Similarly, when all the process is done, we can see the final stable matching results are \( (E_1, N_3), (E_2, N_5), (E_3, N_2), (E_4, N_1), \) and \( (E_5, N_4) \).

Whereas, for the matching in Fig. 3(b) with incomplete preference list, the doubtless stable pairs are \( (E_1, N_2), (E_2, N_3) \), and \( (E_4, N_5) \). As for the unpaired individuals, i.e. \( E_3, E_5, N_1, \) and \( N_4 \), there exist two possibilities to combine them, i.e., \( (E_3, N_5) \) and \( (E_5, N_4) \) or \( (E_3, N_4) \) and \( (E_5, N_1) \). However, according to Definition 2, we can see that both of the options belong to stable matching. Therefore, for HR problem with the incomplete preference list, one important conclusion is that the stable matching can be partial. Besides, it has been proved that for HR with incomplete preference list problems, there may be more than one stable matching, but their size is...
all the same and one of them can be obtained in poly time [30] [31]. In the next subsection, 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 constrain in practice, edge nodes can only choose those participants whose remaining battery capacity is larger than the pre-defined threshold $\delta$. Therefore, the acceptable participants set for edge nodes $E_j$ ($\forall E_j \in E$) can be defined as

$$A(E_j) = \{N_i \in N \mid \delta < c_i\}. \quad (14)$$

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

$$A(N_i) = \{E_j \in E\}. \quad (15)$$

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 $E_j$ can be established based on the computation time $T_i^{cmp}$, where $T_i^{cmp}$ indicates the least time consumption. Correspondingly, we can define the preference list of edge node $E_j$ as

$$L(E_j) = T_i^{cmp}, \forall N_i \in A(E_j). \quad (16)$$

Obviously, $L(E_j)$ 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

$$L(N_i) = T_{ij}^{cmp}, \forall E_j \in A(N_i). \quad (17)$$

$L(N_i)$ is in an ascending order because less communication time is preferable in accordance with a higher channel gain.

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. Firstly, 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 $L(N_i)$. If the number of selected participants exceeds edge node capacity, edge node $E_j$ will only keep those desirable participants within capacity and reject the applications of the others. Then, the unmatched

<table>
<thead>
<tr>
<th>Algorithm 1 Participants and Edge Nodes Pairing with Incomplete Preference List</th>
</tr>
</thead>
<tbody>
<tr>
<td>1: Input: participant set $N$, edge node set $E$, remaining battery energy for each participant $c_i$, energy threshold $\delta$, participant CPU frequency $f_i$, channel gain $h_{ij}$, edge node capacity $q_j$.</td>
</tr>
<tr>
<td>2: For all participants, check their remaining battery capacity.</td>
</tr>
<tr>
<td>3: For all edge nodes, build their preference lists.</td>
</tr>
<tr>
<td>4: For $i=1:N$ do</td>
</tr>
<tr>
<td>5: Build the preference list $L(N_i)$;</td>
</tr>
<tr>
<td>6: end for</td>
</tr>
<tr>
<td>7: For all edge nodes, build their preference lists.</td>
</tr>
<tr>
<td>8: For $j=1:E$ do</td>
</tr>
<tr>
<td>9: Build the preference list $L(E_j)$;</td>
</tr>
<tr>
<td>10: end for</td>
</tr>
<tr>
<td>11: while $\exists E_j \in E$ is available and edge node has a non-empty preference list do</td>
</tr>
<tr>
<td>12: //Participants propose to match with edge nodes</td>
</tr>
<tr>
<td>13: for Unmatched $N_i \in N$ do</td>
</tr>
<tr>
<td>14: Propose to the top-ranked edge node in participant $N_i$’s preference list $L(N_i)$;</td>
</tr>
<tr>
<td>15: Remove the top-ranked edge node from participant $N_i$’s preference list $L(N_i)$;</td>
</tr>
<tr>
<td>16: end for</td>
</tr>
<tr>
<td>17: //Edge node overflow notification</td>
</tr>
<tr>
<td>18: for $E_j \in E$ do</td>
</tr>
<tr>
<td>19: if The number of candidates exceeds capacity $q_j$ then</td>
</tr>
<tr>
<td>20: Hold $q_j$ participants based on its preference list $L(E_j)$ and inform the other participants that they are get rejected from $E_j$.</td>
</tr>
<tr>
<td>21: else</td>
</tr>
<tr>
<td>22: Hold all the participants.</td>
</tr>
<tr>
<td>23: end if</td>
</tr>
<tr>
<td>24: end for</td>
</tr>
<tr>
<td>25: end while</td>
</tr>
<tr>
<td>26: //Partial matching checking</td>
</tr>
<tr>
<td>27: for Unmatched $N_i \in N$ do</td>
</tr>
<tr>
<td>28: Randomly assign them to the available edge node.</td>
</tr>
<tr>
<td>29: end for</td>
</tr>
<tr>
<td>30: Output: Stable many-to-one matching results.</td>
</tr>
</tbody>
</table>

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 is illustrated in Fig. 3. Therefore, when the matching arrives 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 the Algorithm 1, the proposed many-to-one matching method is able to converge and obtain stable
TABLE II

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

matching results. Besides, as for the performance, the running time of implementation can achieve $O(N \times E)$, where $(N \times E)$ denotes all the possible matching pairs.

Proof 1: Let $\mathcal{M}$ be an instance of HR matching. The stable pairs in $\mathcal{M}$ can be found in $O(N)$ time. And the stable matchings in $\mathcal{M}$ can be listed in $O(E)$ time per matching, after $O(N)$ pre-processing time. Therefore, the overall running time can achieve $O(N \times E)$. 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 incomplete preference list (IPL) with regard to system latency. Meanwhile, we take matching with the complete preference list (CPL) and random matching as baseline methods for 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 20MHz. 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 23dBm, 1,000, 50, and 5MB, respectively. The Gaussian noise $N_0$ is set as -96dBm. The energy threshold for every participant is assumed the same as well, which is set as 40%. 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 $\epsilon$ 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.

Firstly, 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. Average network latency with different number of users.

Fig. 4. We increase the number of participants from 1,000 to 10,000 by step 1,000 to show the change of system latency. Among the three methods shown in Fig. 4, 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 higher channel gain for 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 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 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 5,000 and the capacity of edge nodes is 1,000. 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, random method generates the worst performance reflecting by the highest latency. When the edge node number is five, CPL achieves 0.1072 second less delay than random method and IPL achieves 0.0533 second less latency than random method. Besides, we can see that with the increase of number of edge nodes, the overall latency goes down, which is understandable. When the number pf edge node is larger, the elements in the acceptable sets $A(N_i)$ and $A(\mathcal{E}_j)$ become larger accordingly, which
things. Therefore, we are able to find a better assignment with the 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 5,000 and the number of edge nodes as 10. The capacity of each edge node is increased from 500 to 1,000 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 second more than CPL method, and 0.0390 second more than 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 1,000 capacity cases are the same. This is because when the edge node capacity is sufficiently large, the matching assignments achieves stable 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.

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 capacity are set as 5,000, 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 random method gives the largest latency. Statistically, random strategy yields 2.5691 seconds more delay than CPL and 1.5724 seconds more latency than IPL. Macroscopically, with a higher accuracy, the latency is much larger as well. Actually, what is affected by local accuracy is computation 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 increases explosively, resulting in 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 5,000, 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. Apparently, random method gives the highest network latency while CPL gives the lowest, which keeps consistent with previous experiments. In the average, random method generates 4.2652 seconds higher latency than CPL and 2.5547 seconds higher latency than IPL. In general, with the increase of 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 to 1, i.e., $N_i$ will join federated learning task. Hence, as energy threshold increases, the total number of participants involving in the learning process will decrease. Therewith, the overall network latency can be reduced correspondingly.

Finally, we adjust the parameter of 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 5,000, 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. And 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 nodes 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 study the low latency problem for multi-task federated learning in the MEC networks. Considering in large scale IoTs scenario, it is not possible for edge nodes and end devices to obtain the complete information of the other side, which means building the complete preference list is impractical. Therefore, we propose a method to deal with the two-sided many-to-one matching with the incomplete preference list. The simulation results show that the performance of our proposed method is close to the performance of CPL, although there is small gap between them due to information missing. Besides, we also 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. 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.

**ACKNOWLEDGMENT**

The research is supported in part by US Multidisciplinary University Research Initiative 18RT0073, in part by NSF EARS-1839818, CNS1717454, CNS-1731424, and CNS-1702850, in part by the National Key Research and Development Program of China (No. 2020YFC1511801), in part by the National Natural Science Foundation of China under Grant No. U2066201 and Grant 61871416, and in part by the Beijing Municipal Natural Science Foundation under Grant L192030.

**REFERENCES**


Dawel Chen (S’19) received the BS degree in telecommunication engineering from the Huzhou University of Science and Technology, Wuhan, China, in 2015. He is currently working toward the PhD degree at the Department of Electrical and Computer Engineering, University of Houston, TX. His research interests include machine learning, cloud/edge computing, and wireless networks.

Choong Seon Hong Choong Seon Hong (S’95–M’97–SM’11) 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, Gyeonggi-do, South Korea, 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. Dr. Hong is a member of the Association for Computing Machinery (ACM), the Institute of Electronics, Information and Communication Engineers (IEICE), the Information Processing Society of Japan (IPSJ), the Korean Institute of Information Scientists and Engineers (KIIEEE), the Korean Institute of Communications and Information Sciences (KICS), the Korean Information Processing Society (KIPS), and the Open Standards and ICT Association (OSIA). He 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 (NOMS), International Symposium on Integrated Network Management (IM), Asia-Pacific Network Operations and Management Symposium (APNOMS), End-to-End Monitoring Techniques and Services (E2EMON), IEEE Consumer Communications and Networking Conference (CCNC), Assurance in Distributed Systems and Networks (ADSN), International Conference on Parallel Processing (ICPP), Data Integration and Mining (DIM), World Conference on Information Security Applications (WISA), Broadband Convergence Network (BnC), Telecommunication Information Networking Architecture (TINA), International Symposium on Applications and the Internet (SAINT), and International Conference on Information Networking (ICON). He was an Associate Editor of the IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT and the IEEE JOURNAL OF COMMUNICATIONS AND NETWORKS and an Associate Editor for 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.

Choong Seon Hong Choong Seon Hong (S’95–M’97–SM’11) 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, Gyeonggi-do, South Korea, 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. Dr. Hong is a member of the Association for Computing Machinery (ACM), the Institute of Electronics, Information and Communication Engineers (IEICE), the Information Processing Society of Japan (IPSJ), the Korean Institute of Information Scientists and Engineers (KIIEEE), the Korean Institute of Communications and Information Sciences (KICS), the Korean Information Processing Society (KIPS), and the Open Standards and ICT Association (OSIA). He 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 (NOMS), International Symposium on Integrated Network Management (IM), Asia-Pacific Network Operations and Management Symposium (APNOMS), End-to-End Monitoring Techniques and Services (E2EMON), IEEE Consumer Communications and Networking Conference (CCNC), Assurance in Distributed Systems and Networks (ADSN), International Conference on Parallel Processing (ICPP), Data Integration and Mining (DIM), World Conference on Information Security Applications (WISA), Broadband Convergence Network (BnC), Telecommunication Information Networking Architecture (TINA), International Symposium on Applications and the Internet (SAINT), and International Conference on Information Networking (ICON). He was an Associate Editor of the IEEE TRANSACTIONS ON NETWORK AND SERVICE MANAGEMENT and the IEEE JOURNAL OF COMMUNICATIONS AND NETWORKS and an Associate Editor for 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.

Li Wang (S’08–M’14–SM’16) received the Ph.D. degree from the 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 heads the High Performance Computing and Networking Laboratory. She is also a member of the Key Laboratory of the Universal Wireless Communications, Ministry of Education, China. She has also held Visiting Positions with the School of Electrical and Computer Engineering, Georgia Tech, Atlanta, GA, USA, from December 2013 to January 2015, and with the Department of Signals and Systems, Chalmers University of Technology, Gothenburg, Sweden, from August to November 2015 and July 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. 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. She was the Symposium Chair of IEEE ICC 2019 on Cognitive Radio and Networks Symposium and a Tutorial Chair of IEEE VTC 2019-fall. She also serves as the vice chair of Meetings and Conference Committee (MCC) for IEEE Communication Society (ComSoc) Asia Pacific Board (APB) for the term of 2020-2021, and chairs the special interest group (SIG) on Social Behavior Driven Cognitive Radio Networks for IEEE Technical Committee on Cognitive Networks. She was the recipient of the 2013 Beijing Young Elite Faculty for Higher Education Award, Best Paper Awards from several IEEE conferences, e.g., IEEE ICC 2017, IEEE GLOBECOM 2018, IEEE WCSP 2019, and so forth. She was also the recipient of the Beijing Technology Rising Star Award in 2018. 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.

Yiyong Zha received the B.S. in applied physics from University of Science and Technology of China in 2008, and the PhD in Electrical and Computer Engineering from University of Houston in 2013. He worked as a researcher in networking and cloud industry after graduation. Dr. Zha has many published papers and patents. Now he serves as an expert engineer in Tencent technology. His research interest is mainly focused on 5G, edge computing and real-time service on cloud.

Yunfei Zhang is the Director of Tencent Future Network Lab. He is also an Adjunct Professor at Beijing Jiaotong University. 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 serves as IEEE GLOBECOM 2019 Keynote speech session and ICC 2019 forum co-chair. He has rich experience in internet and wireless standards, established and chaired the working group of P2P Streaming Media Protocol (PPSP) in IETF, served as ITU-T editor, led the first 3GPP standard team in Chinese smart phone vendors, and issued more than 10 international standards in IETF, ITU-T and etc.

Xin Liu has more 20 years of R&D experience in internet and communication industry. Start from 3G to 5G, he actively leading the technology platform RD team in Tencent. He has many publications and patents on mobile internet and communication industry. Now he is the department manager of Tencent PCG (Platform & Content Group). He is leading the future network team that is contributing to standard bodies such as 3GPP, ITU and IETF. Also he serves as a director of the Linux Foundation and committing to open source collaboration in the industry.

Zhu Han (S’01–M’04–SM’09–F’14) received the B.S. degree in electronic engineering from Tsinghua University, in 1997, and the M.S. and Ph.D. degrees in electrical and computer engineering from the University of Maryland, College Park, in 1999 and 2003, respectively.

From 2000 to 2002, he was an R&D Engineer of JDUS, Germantown, Maryland. From 2003 to 2006, he was a Research Associate at the University of Maryland. From 2006 to 2008, he was an assistant professor at Boise State University, Idaho. Currently, he is a John and Rebecca Moores Professor in the Electrical and Computer Engineering Department as well as in the Computer Science Department at the University of Houston, Texas. 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 an 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, IEEE Leonard G. Abraham Prize in the field of Communications Systems (best paper award in IEEE JSAC) in 2016, and several best paper awards in IEEE conferences. Dr. Han was an IEEE Communications Society Distinguished Lecturer from 2015-2018, AAAAS fellow since 2019 and ACM distinguished Member since 2019. Dr. Han is a highly cited researcher since 2017 according to Web of Science. Dr. Han is also the winner of 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.”