Distributed Edge Cooperation and Data Collection for Digital Twins of Wide-Areas

Mancong Kang, Xi Li[*], Hong Ji, Heli Zhang

Key Laboratory of Universal Wireless Communications, Beijing University of Posts and Telecommunications, Beijing 100876, China
[*] The corresponding author, email: lixi@bupt.edu.cn

Cite as: M. Kang, X. Li, *et al.*, "Distributed edge cooperation and data collection for digital twins of wide-areas," *China Communications*, vol. 20, no. 8, pp. 177-197, 2023. **DOI**: 10.23919/JCC.fa.2023-0202.202308

Abstract: Digital twins for wide-areas (DT-WA) can model and predict the physical world with high fidelity by incorporating an artificial intelligence (AI) model. However, the AI model requires an energy-consuming updating process to keep pace with the dynamic environment, where studies are still in infancy. To reduce the updating energy, this paper proposes a distributed edge cooperation and data collection scheme. The AI model is partitioned into multiple sub-models deployed on different edge servers (ESs) co-located with access points across wide-area, to update distributively using local sensor data. To reduce the updating energy, ESs can choose to become either updating helpers or recipients of their neighboring ESs, based on sensor quantities and basic updating convergencies. Helpers would share their updated sub-model parameters with neighboring recipients, so as to reduce the latter updating workload. To minimize system energy under updating convergency and latency constraints, we further propose an algorithm to let ESs distributively optimize their cooperation identities, collect sensor data, and allocate wireless and computing resources. It comprises several constraint-release approaches, where two child optimization problems are solved, and designs a large-scale multi-agent deep reinforcement learning algorithm. Simulation shows that the proposed scheme can efficiently reduce updating energy compared with the baselines.

Keywords: digital twin; smart city; multi-agent deep reinforcement learning; resource allocation

Received: Mar. 22, 2023
Revised: May. 24, 2023
Editor: Guanglin Zhang

I. INTRODUCTION

Digital twin (DT) is a cutting-edge technology that creates a digital replica of a physical entity based on advanced modeling technologies and real-time physical data [1]. With the advent of 6G, there has been a call to establish DTs for geographically wide-areas ("DT-WA") [2], which can fuse the physical and digital worlds to facilitate large-scale monitoring, pre-validation, prediction, and intelligent management in various fields, such as transportation, water resources management, city planning, building construction and emergency orchestration [3]. For instance, DT-WAs have been built to forecast city flood [4] and forest fire [5]. According to the International Data Corporation, the global investment in DT-WA is expected to reach \$107.826 billion by 2025, with a compound annual growth rate of 34.13% over five years.

A critical component of DT-WA is the behavior model of the physical world, which can be used to pre-validate proposals and predict future states for advanced management. In particular, it is expected to be realized using an artificial intelligence (AI) model [6], which would be established in a distributed form with a large quantity of sub-models deployed on different edge servers, co-located with different access points (APs), so as to capture various locations in the wide-area. Specifically, the AI model needs to incorporate a distributed updating process to maintain its predicting and simulation accuracy for the dynamic environment. That is, sub-models on different ESs should regularly update their parameters, more precisely the integrated learning networks (e.g., deep neural networks), based on the collected local sensor data. However, due to the extensive coverage of DT-WA, the updating process of the whole AI model would consume a massive amount of energy. Nevertheless, the study of the updating process is still in its early stages, despite its great potential for wide application.

To address the energy issues in the updating process, we identify six distinct features in the AI model of DT-WA:

- Sub-models quantity: The AI model consists of a large number of sub-models on different ESs to model every part of the wide-area. For example, a DT-WA for city flood predictions requires to deploy sub-models in various locations such as different river banks, buildings, roads, and mountain areas [4].
- Sub-models heterogeneous: Sub-models on different ESs would be heterogeneous. Their structures and learning goals may be different depending on their local features. For example, the sub-models on ESs connected with APs in mountains may need to predict the future soil moistures, while river banks predict the water levels.
- Sensors heterogeneous: The updating process relies data from various kinds of sensors, such as monitors and thermo-hygrometers. They may have distinct communication requirements, such as enhanced mobile broadband (eMBB) and ultra-reliable low latency communications (URLLC). Furthermore, different sub-models may rely on different types of sensors at varying levels.
- Sensors uneven availability: The distribution of heterogeneous sensors across different ESs is not uniform. Moreover, their availabilities are dynamically changing, determined by their current battery levels which are influenced by the sensor activations, transmitting powers, and the speeds of solar charging. Consequently, the size of collected local datasets on different ESs may vary, inducing dissimilar sub-model updating convergencies and energies.
- Neighboring similarity: Sub-models on neighboring ESs may exhibit similar updating patterns, due to the ensemble change of system environment. For example, sub-models for adjacent river banks and mountain areas may have partially similar model architectures to infer future water level or soil moisture based on the present air humid and monitoring image. It offers an opportunity for neighbors to partially share their updating parameters.
- Community benefit: Each ES needs to act on the basis of community benefit. Individual sacrifice is encouraged to minimize the whole system cost. It is different from traditional multi-users/ESs scenarios [7], where each entity wants to minimize its own cost while achieving a good social welfare.

Based on the above features, this paper proposes a distributed edge cooperation and data collection scheme to reduce the overall updating energy of DT-WA. Considering the sub-models on adjacent ESs may exhibit similar updating trends, each ES can choose to become either an updating helper or a recipient of its neighboring ESs, based on the quantities of available sensors, environmental changes, and basic updating convergencies. ESs with more available sensors are more likely to become helpers to reduce the system energy, since larger collected datasets can assist sub-models to achieve higher updating convergency with lower computing energy [8]. Each helper first activates some of its local heterogeneous sensors to collect sensor data and update its sub-model, then sends the updated sub-model parameters to its neighboring recipient ESs. Each recipient combines the useful part from its helper's parameters with its local sub-model to increase its basic updating convergency. Then, the recipient ES only needs to further update its sub-model based on local sensor data to capture local features with reduced workload. This approach enables a helper ES to save the updating energy of several neighboring recipient ESs, which can reduce the overall system energy.

Moreover, to facilitate the ESs decision making process, we investigate a cooperation and resource allocation problem, with the aim to minimize the long-term system energy under high updating convergencies and low latencies. It is complex due to three reasons. Firstly, each ES's action can have a ripple effect on many other ESs in the network, making it challenging for each ES to make optimal decisions with only local information. This is compounded by the large number of ESs with a random topology. Secondly, the activations of different types of sensors must consider their different impacts on the local sub-model, their availabilities and their influences on each other during wireless transmission process. Thirdly, the dynamic environment leads to varying updating similarities among neighboring ESs and dynamic updating workload for each sub-model. Meanwhile, the sensor battery levels are constantly changing. Each ES must consider not only the current situation but also the historical and potential future conditions when cooperating, activating sensors, and allocating resources.

In view of the above issues, we propose a distributed cooperation and resource allocation algorithm. First, the problem is reformulated as a decentralized partially observable Markov decision process (Dec-POMDP), where each ES needs to dynamically decide its cooperation identity, activate heterogeneous sensors, and allocate wireless and computing resources, based on its surrounding observations. Then, to reduce the problem complexity, we solve two child optimization problems to minimize the local computing energy on all ESs and the surrounding energy on helper ESs, respectively. After that, we design a distributed large-scale multi-agent proximal policy optimization algorithm to solve the Dec-POMDP. Simulation results show that the proposed scheme can effectively reduce the DT-WA updating energy compared with the baselines.

The contribution can be summarized as follows:

- The paper closes the gap of studying the updating process for AI model in DT-WA, and designs a cooperation and data collection scheme for it to reduce its overall updating energy.
- An optimization problem is carefully established, where ESs needs to dynamically adjust their cooperative identities, activate heterogenous sensors and allocate wireless and computing resources, with the aim to minimize the average long-term overall updating energy, under updating convergency and latency requirement.
- We propose a distributed algorithm to solve the problem. The original problem is transformed into a Dec-POMDP. Then, the Dec-POMDP is simplified by solving two child optimization problems. Finally, distributed large-scale multi-agent proximal policy optimization algorithm is proposed for each ESs to individually optimize its action.
- Simulation shows that the proposed edge cooperation and data collection scheme can efficiently reduce DT-WA updating energy under high updating convergency and low latency compared with the baselines.

The remainder of our work is organized as follows. Section II gives the related works. Section III and Section IV gives the system model and problem formulation, respectively. Section V designs an algorithm to solve the problem. Simulation results and discussions are given in Section VI. Finally, we conclude this paper in Section VII.

II. RELATED WORKS

This section briefly summarizes the existing works of DTs in wireless networks, and the previous works on DT-WA.

2.1 Digital Twins in Wireless Networks

Existing researches in wireless communication field mainly studied DT for services and wireless networks. On the one hand, researchers have proposed several updating schemes for service DTs. They normally leveraged the federated learning (FL) to learn the common feature in different devices with similar DT model structures, to obtain global DT model for specific services (e.g., traffic predictions) [9, 10]. For instance, authors in [9] proposed an asynchronous model update scheme to efficiently aggregate the global service DT on base station (BS) based on the local DTs trained on different IoT devices. On the other hand, researchers have leveraged network DTs to capture the real-time network states for resource optimizations [11, 12]. For example, authors in [12] proposed a DT-assisted task offloading scheme, where BS can make optimal offloading decisions based on the network DT by obtaining the state of computation resource on different user devices. They rarely studied the updating process for network DTs, since network DTs are mainly consisted of DTs of man-made network elements and mobile devices [13]. These DTs can be provided by manufacturers, which are comparatively more stable, controllable and predictable than the nature environment in DT-WA, therefore do not need frequent updating process.

In summary, service DTs usually assume similar model structures within networks, whose FL-based updating strategies can not be applied to DT-WA to update the heterogeneous sub-models in DT-WA. Moreover, network DTs do not need such frequent updating process as DT-WA. Therefore, their strategies are not applicable for DT-WA.

2.2 Digital Twins of Wide-areas

DT-WA has two basic functions, monitoring and modeling the physical world. On the one hand, to realize the monitoring function, researchers have explored various three-dimensional (3D) reconstructing technologies to highly restore the 3D state of the physical world in digital space [3]. Others developed efficient data synchronizing schemes to collect massive physical data via wireless networks to synchronize the digital counterpart [14]. On the other hand, to realize the modeling function, researches have explored various modeling technics for DT-WA [15]. In particular, the deep learning algorithm has been seen as one of the most prevail modeling approaches [6, 16]. For instance, authors in [16] combined semantic knowledge with deep neural network (DNN) to presenting and reasoning the DT-WA, so as to identify events and make decision automatically.

However, the study for updating the AI model in DT-WA is still in its infancy. It is critical for the pre-validation and predicting performance in DT-WA, which has a wide potential use in various fields. To close the gap, this paper studies the updating process, and solves its energy challenge by designing an energy-efficient edge cooperation and data collection scheme.

III. SYSTEM MODEL

3.1 Preliminaries

Large quantities of ESs co-located with different APs are distributed across the wide-area, which are denoted by a set $\mathcal{F} = (1, 2, \cdots, F)$. A DT-WA is established for the system, whose AI model is denoted by $\mathcal{M}$. It is divided into multiple local sub-models $\{\mathcal{M}_i | i \in \mathcal{F}\}$ deployed on different ESs for distributed updating, as depicted in Figure 1. Distance between two APs is $d_{i,j}$ $(i, j \in \mathcal{F})$. The set of neighboring ESs of ES i is denoted by $\mathcal{B}_i = \{j | d_{i,j} < d_{\text{th}}, j \in \mathcal{F}\}$, where d_{th} is the distance threshold between their connected APs. ESs can share their updated sub-model parameters with their neighbors to save each others energy in each updating circle. The solar-powered heterogeneous sensors under ES i include eMBB sensors and URLLC sensors, which are denoted by two sets $\mathcal{D}_i^{\text{B}} = \{1, 2, \cdots, N_i^{\text{B}}\}$ $(i \in \mathcal{F})$ and $\mathcal{D}_i^{\text{U}} = \{1, 2, \cdots, N_i^{\text{U}}\}$ $(i \in \mathcal{F})$, where N_i^{B} and N_i^{U} are the numbers of different type of sensors, respectively. Time is discretized into updating circles, which is denoted by $\mathcal{T} = \{1, 2, \cdots, t, \cdots, +\infty\}$.

3.2 Dataset Collecting Process

In each updating circle, each ES activates part of its available heterogenous sensors and collects sensor data through wireless communication. The collected data is used for updating its local sub-model $\mathcal{M}_i$. The heterogenous sensors are divided into URLLC-type and eMBB-type depending on their transmitted data-types, where we use superscript 'B' and 'U' to distinguish their variables, respectively. The multiplexing method for mixed eMBB and URLLC transmission is based on the well-known puncture approach [17]. Specifically, each updating circle are divided into K time slots, denoted by $\{1, \cdots, k, \cdots, K\}$. Each slot is further divided into 10 mini-slots with a length of T_{s} (0.1ms). The eMBB-type sensors continuously transmitting its dataset within allocated bandwidth, whereas the URLLC packets can pre-emptively overlap the eMBB traffic in each mini-slot.

3.2.1 URLLC-Type Sensors

To avoid co-channel interference, different ESs use orthogonal frequencies to collect datasets from its local sensors. The bandwidth used by each ES is B. We use a binary variable $v_{i,n}^{\text{U}}(t)$ to denote whether a URLLC-type sensor n under ES i connected AP is available to activate, i.e., having enough energy to work. If $v_{i,n}^{\text{U}}(t) = 1$, it is available, and vise versa. We use a binary variable $g_{i,n}^{\text{U}}(t)$ to denote whether sensor n is activated. If $g_{i,n}^{\text{U}}(t) = 1$, it is activated, and vise versa.

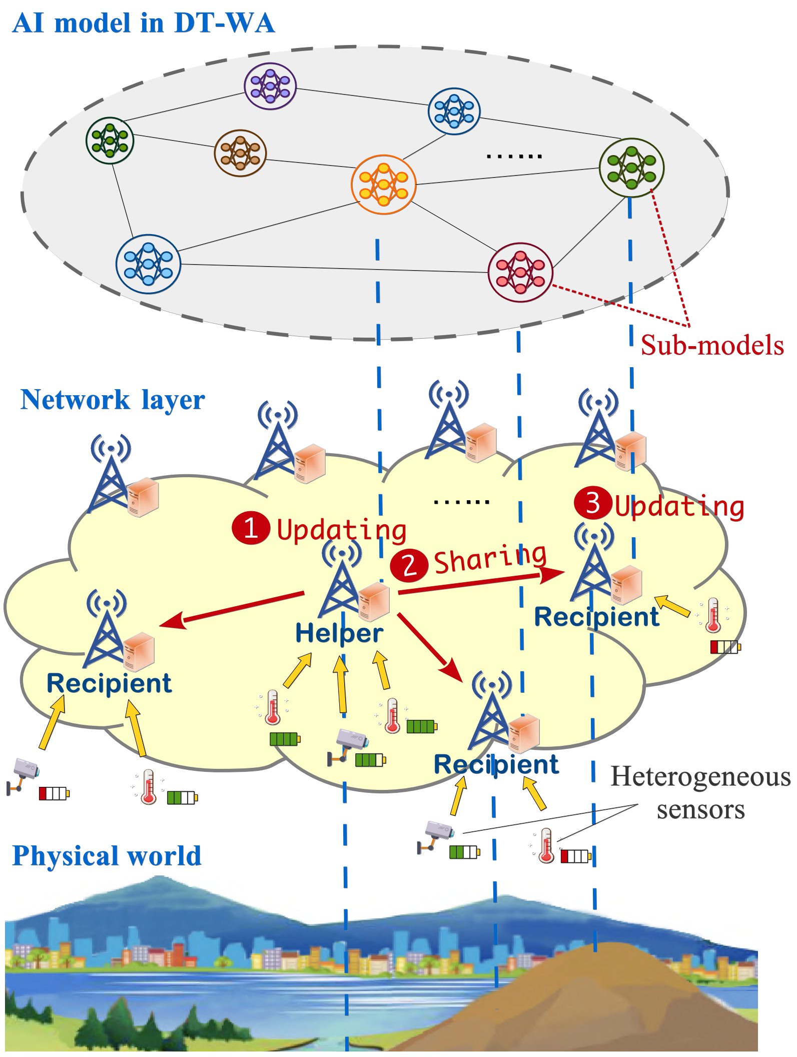


Figure 1. *Distributed edge cooperation and data collection for updating the AI model in DT-WA.*

Only available sensors can be activated, i.e.,

$$g_{i,n}^{\mathrm{U}}(t) \leq v_{i,n}^{\mathrm{U}}(t). \tag{1}$$

It spontaneously generates and sends datasets to the local ES, with a probability of q in each mini-slot. The packet size in one transmission is:

$$S^{\mathrm{U}} = s^{\mathrm{U}} \cdot l^{\mathrm{U}}, \tag{2}$$

where l^{U} is the number of samples in a dataset, s^{U} is the bit size of one sample. Its transmission rate can no longer achieve Shannon capacity because of the finite-blocklength, where the instantaneous achievable rate is modified into [18]:

$$R(t,k) \approx \frac{B}{\ln 2}\left[\ln\left(1+\gamma(t,k)\right) - \sqrt{\frac{V(t,k)}{T_s B}} f_{\mathrm{Q}}^{-1}(\varepsilon(t,k))\right], \tag{3}$$

where N_0 is the noise spectral density, $\gamma(t,k) = H(t,k)P_{i,k}^{\mathrm{U}}(t,k)/(N_0 B)$ is the received SNR, $H(t,k)$ is the instant channel gain, $V(t,k) = 1 - [1+\gamma(t,k)]^{-2}$, $f_{\mathrm{Q}}^{-1}(\cdot)$ is the inverse function of the Q-function, $\varepsilon(t,k)$ is the decoding error probability. $P_{i,n}^{\mathrm{U}}$ is the fixed transmitting power of the n-th URLLC sensor under ES i. It can not adapt to the instant channel gain because the CSI cannot be applied to the

URLLC transmission. Different URLLC sensors determine their transmitting power $P_{i,n}^{\mathrm{U}}$ according to [19] to achieve decoding error probability $\varepsilon_{\mathrm{th}}^{\mathrm{U}}$ under transmission rate $R^{\mathrm{U}} = S^{\mathrm{U}}/T_{\mathrm{s}}$.

3.2.2 EMBB-Type Sensors

We use a binary variable $v_{i,n}^{\mathrm{B}}(t)$ to denote whether a eMBB-type sensor n under ES i is available to activate, i.e., having enough energy to work. If $v_{i,n}^{\mathrm{B}}(t) = 1$, it is available, and vise versa. We use a binary variable $g_{i,n}^{\mathrm{B}}(t)$ to denote whether sensor n is activated. If $g_{i,n}^{\mathrm{B}}(t) = 1$, it is activated, and vise versa. Only available sensors can be activated, i.e.,

$$g_{i,n}^{\mathrm{B}}(t) \leq v_{i,n}^{\mathrm{B}}(t). \tag{4}$$

The activated eMBB sensors equally divide the bandwidth of ES i. Therefore, according to the Shannon equation, the wireless transmission rate of each sensor under ES i is

$$\begin{aligned} R_i^{\mathrm{B}}(t) = & \\ & (1 - q_i^{\mathrm{sum}}(t))B_i^{\mathrm{s}}(t)\log_2\left(1 + \frac{H(t,k)p_{i,n}^{\mathrm{B}}(t,k)}{N_0 B_i^{\mathrm{s}}(t)}\right) \\ & = (1 - q_i^{\mathrm{sum}}(t))B_i^{\mathrm{s}}(t)\cdot\log_2\left(1 + \frac{P_i^{\mathrm{B,r}}(t)}{N_0 B_i^{\mathrm{s}}(t)}\right), \end{aligned} \tag{5}$$

where $B_i^{\mathrm{s}}(t) = B/(\sum_{n=1}^{N_i^{\mathrm{B}}} g_{i,n}^{\mathrm{B}}(t))$, $q_i^{\mathrm{sum}}(t) = 10q\sum_{n=1}^{N_i^{\mathrm{U}}} g_{i,n}^{\mathrm{U}}(t)/T_{\mathrm{s}}$ is the ratio of time-resource occupied by the URLLC sensors. For simplicity, different eMBB sensors under ES i achieve the same target receiving power $P_i^{\mathrm{B,r}}(t)$ in the t-th updating circle, by adapting their instant transmitting powers according to the instant channel gain under given wireless decoding error probability $\epsilon_{\mathrm{th}}^{\mathrm{B}}$ with the power scheme in [20].

3.3 Cooperation and Updating Process

3.3.1 ESs Cooperation Process

During an updating circle, first, each ES chooses its cooperative identity (helper or recipient), which uses a binary variable $m_i(t)$ to convey. $m_i(t) = 1$ denotes the ES chooses to be a helper, while $m_i(t) = 0$ denotes a recipient. All of the ESs that choose to be helpers are gathered into a helper set $\mathcal{H}(t) = \{i | m_i(t) = 1, i \in \mathcal{K}\}$, which is unknown to any ES. Similarly, the ESs that choose to be recipients are gathered into a recipient set $\mathcal{O}(t) = \{i | m_i(t) = 0, i \in \mathcal{K}\}$. Then, all of ESs broadcast their identities to their neighboring ESs through wired link between APs, so that each recipient know the helpers in its neighborhoods. The recipient would randomly choose a helper from its neighborhoods, which is denoted by

$$o_i(t) = \begin{cases} j, & j \in \mathcal{B}_i^*, \text{ if } \mathcal{B}_i^* \neq \emptyset, \\ \text{null}, & \text{if } \mathcal{B}_i^* = \emptyset, \end{cases} \tag{6}$$

where $i \in \mathcal{O}(t)$ and $\mathcal{B}_i^* = \mathcal{B}_i \cap \mathcal{H}(t)$. They would send an informing message to their helpers. Then, each helper is aware of its recipients, which can be represented by a set

$$\mathcal{G}_i(t) = \{j | o_j(t) = i, j \in \mathcal{O}(t)\}, i \in \mathcal{H}(t). \tag{7}$$

After that, helper i ($i \in \mathcal{H}(t)$) updates its sub-model $\mathcal{M}_i$ based on its local data. Then, it shares the updated sub-model parameters with its recipients through wired channels with a negligible latency. Each recipient j ($j \in \mathcal{G}_i(t)$) extracts the useful part in $\mathcal{M}_i$, and incorporates it into its old sub-model $\mathcal{M}_j$ to increase its basic updating convergency. Then, the recipient further updates its sub-model $\mathcal{M}_j$ based on its local data.

3.3.2 Sub-Model Local Updating Process

Helper ESs would start to update their local sub-model based on their heterogenous sensor data after their helper identities have been determined. Recipient ESs start to update their local sub-model based on sensor data after receiving their helpers' sub-model parameters. For the local sub-model on ES i, its learning network (commonly a deep neural network) can be divided into two child networks that can be independently updated using eMBB and URLLC dataset, which are later integrated into an entire learning network using mixed eMBB-URLLC dataset to further update.

IV. PROBLEM FORMULATION

Our goal is to let the ESs individually make decisions, e.g., choosing cooperative identity, to update their local sub-models in DT-WA under updating convergency and latency requirements, with the aim to minimize the long-term system energy. Next, we give the mathmetical relationships between the decision variables with different performance metrics (i.e., updating convergency, latency, system energy). Then, we give our optimization problem.

4.1 Updating Latency

At the beginning of each updating circle, each ES chooses to be a helper or recipient, and decides to activate local heterogeneous sensors. Then, they begin to collect the local data from their local sensors. After that, each helper ES updates local sub-model based on the local sensoring data. Then, it sends the updated sub-model to its recipient ESs. Each recipient partly combine its helper sub-model with its old sub-model and further updates it using local dataset.

First, we give the heterogeneous data collecting delay of all ES. Considering the URLLC packets puncture the eMBB data streams in the wireless transmission process, the transmission delay of heterogeneous data is equal to the eMBB transmission delay. Besides, each ES would determine the received power for its local eMBB sensors, which is in the same value for different sensors. It leads to a same transmission rate $R_i^{\mathrm{B}}(t)$ of all eMBB sensors under ES i, according to Eq. (5). The transmission delay would be

$$D_i^{\mathrm{t}}(t) = \frac{s^{\mathrm{B}} \cdot l^{\mathrm{B}}}{R_i^{\mathrm{B}}(t)}, \tag{8}$$

where l^{B} is the number of samples in a dataset generated by a eMBB sensor, s^{B} is the bit size of one sample.

Second, we give the local updating delay of all ES, where each ES uses computing resources and collected data to update its sub-model parameters. On ES i, the computing workload of its sub-model M_i comprises three parts, which are eMBB and URLLC-data related child network updating and integrated updating, given as

$$\begin{cases} C_i^{\mathrm{B}}(t) = I_i^{\mathrm{B}}(t) \cdot Q_i^{\mathrm{B}}(t) \cdot c^{\mathrm{B}}, \\ C_i^{\mathrm{U}}(t) = I_i^{\mathrm{U}}(t) \cdot Q_i^{\mathrm{U}}(t) \cdot c^{\mathrm{U}}, \\ C_i^{\mathrm{Int}}(t) = I_i^{\mathrm{Int}}(t) \cdot \min\{Q_i^{\mathrm{B}}(t), Q_i^{\mathrm{U}}(t)\} \cdot c^{\mathrm{Int}}, \end{cases} \tag{9}$$

where $c^{\mathrm{B}}, c^{\mathrm{U}}, c^{\mathrm{Int}}$ are the computing workloads of one piece of data sample in different child network or the integrated learning network, $I_i^{\mathrm{B}}(t)$, $I_i^{\mathrm{U}}(t)$ and $I_i^{\mathrm{Int}}(t)$ are training iterations in different sub-processes, which are determined by resource optimization algorithm on each ES. $Q_i^{\mathrm{B}}(t) = l \cdot \sum_{n=1}^{N_i^{\mathrm{B}}} g_{i,n}^{\mathrm{B}}(t)$, $Q_i^{\mathrm{U}}(t) = l \cdot \sum_{n=1}^{N_i^{\mathrm{U}}} g_{i,n}^{\mathrm{U}}(t)$ are the size of datasets in different sub-processes. Then, the whole computing workload is

$$C_i(t) = C_i^{\mathrm{B}}(t) + C_i^{\mathrm{U}}(t) + C_i^{\mathrm{Int}}(t). \tag{10}$$

The computing latency on ES i is

$$D_i^{\mathrm{c}}(t) = C_i(t)/f_i(t), \tag{11}$$

where $f_i(t)$ is the allocated computation resource on ES i.

Based on the above data collecting and ES computing delay, the overall updating latency of sub-model on every ES, including helpers and recipients, can be calculated by (12).

$$D_i(t) = \begin{cases} D_i^{\mathrm{t}}(t) + D_i^{\mathrm{c}}(t), \text{ if } i \in \mathcal{H}(t) \text{ or } o_i(t) = \emptyset, \\ \max\{D_i^{\mathrm{t}}(t), D_{o_i(t)}(t)\} + D_i^{\mathrm{c}}(t), \text{else}. \end{cases} \tag{12}$$

In (12), the first line gives the overall updating delay of each helper ES and the recipient ES without matched helpers, the second line gives the delay of each recipient ES. The sub-model transmission delay between helper ES and its recipient ES can be omitted because of large wired transmission capacity.

4.2 Updating Convergency

In the distributed DT-WA model updating scenario, we study the updating convergency of the sub-model on each ES. In each updating circle, due to the dynamic environment changes, the sub-model on an ES may not be able to faithfully replicate the behavior model of its physical environment. The mismatch is reflected in the basic updating convergency $(1-\exp(u_i^{\mathrm{base}}(t)))$, where $u_i^{\mathrm{base}}(t))$ is the basic convergency factor. It is influenced by both the updating convergency in the last updating circle and the volume of environment changes after the latest updates. In particular, for recipient ESs, the basic convergency factor is also influenced by its helper sub-model, which could increase their original values. Commonly, the transition function of the basic convergency factor is unknown, due to the unstructured environment changes.

In the local computing process of every ES, helper or recipient, the updating convergency depends on the sub-model structure, dataset dimension, and training iteration times. In detail, the updating convergency is jointly determined by the three sub-processes, including eMBB and URLLC-related child network updating and integrated updating, as given in Eq. (13).

$$\varepsilon_i(t) = 1 - \underbrace{e^{-u_i^{\mathrm{base}}(t)}}_{\text{Basic divergency.}} \cdot \underbrace{e^{-k_i^{\mathrm{B}} I_i^{\mathrm{B}}(t)\cdot[Q_i^{\mathrm{B}}(t)]^v}}_{\text{EMBB child network.}} \cdot \underbrace{e^{-k_i^{\mathrm{U}} I_i^{\mathrm{U}}(t)\cdot[Q_i^{\mathrm{U}}(t)]^v}}_{\text{URLLC child network.}} \cdot \underbrace{e^{-k_i^{\mathrm{Int}} I_i^{\mathrm{Int}}(t)\cdot[\min\{Q_i^{\mathrm{B}}(t),Q_i^{\mathrm{U}}(t)\}]^v}}_{\text{Integrated network.}}. \tag{13}$$

In the Eq. (13), the convergency performance of each sub-process [8] is jointly determined by dataset dimension $Q_i^{\mathrm{B}}(t)$, $Q_i^{\mathrm{U}}(t)$ and $Q_i^{\mathrm{Int}}(t) = [\min\{Q_i^{\mathrm{B}}(t), Q_i^{\mathrm{U}}(t)\}]$, and training iteration times $I_i^{\mathrm{B}}(t)$, $I_i^{\mathrm{U}}(t)$ and $I_i^{\mathrm{Int}}(t)$. $k_i^{\mathrm{B}}, k_i^{\mathrm{U}}, k_i^{\mathrm{Int}}$ indicates different degrees of importance of different types data in sub-model i. v is determined by the objective functions in sub-models.

4.3 ES Computing Energy

Each ES computing energy is determined by the computing resources $f_i(t)$, computing workload $C_i(t)$ and computing time $D_i^{\mathrm{c}}(t)$ in the local updating process, given by [21]

$$E_i(t) = \kappa \cdot f_i^3(t) D_i^{\mathrm{c}}(t) = \kappa \cdot [C_i(t)]^3/[D_i^{\mathrm{c}}(t)]^2, \tag{14}$$

where κ is the effectively switched capacitor coefficient. In the t-th updating circle, the system energy consumption is the sum of all ES computing energy

$$E(t) = \sum_{i=1}^{F} E_i(t). \tag{15}$$

4.4 Sensor Transition Functions

We give the transition functions of the availability and remaining energy for URLLC and eMBB sensors.

When an URLLC sensor is activated, its consumed energy in this updating circle is a fixed value $(P_{i,n}^{\mathrm{U}} \cdot q)$. Therefore, the remaining energy of a URLLC sensor n in the beginning of the $(t+1)$-th updating circle is

$$\begin{aligned} E_{i,n}^{\mathrm{U}}(t+1) = [E_{i,n}^{\mathrm{U}}(t) - g_{i,n}^{\mathrm{U}}(t) \cdot P_{i,n}^{\mathrm{U}} \cdot q]^+ \\ + E_{i,n}^{\mathrm{U,r}}(t), \end{aligned} \tag{16}$$

where $E_{i,n}^{\mathrm{U,r}}(t)$ is the amount of solar energy harvested in the present updating circle, which is unknown at the beginning of the updating circle when ES makes cooperating decisions. If the remaining energy is larger than a working threshold, then it is available to use in the next updating circle. That is

$$v_{i,n}^{\mathrm{U}}(t+1) = \begin{cases} 1, & \text{if } E_{i,n}^{\mathrm{U}}(t+1) \geq E_{\mathrm{i,n,th}}^{\mathrm{U}}, \\ 0, & \text{else,} \end{cases} \tag{17}$$

where the working threshold $E_{\mathrm{i,n,th}}^{\mathrm{U}}$ is equal to the consumed energy when it is activated $E_{\mathrm{i,n,th}}^{\mathrm{U}} = P_{i,n}^{\mathrm{U}} \cdot q$.

The remaining energy of an eMBB sensor n in the beginning of the next updating circle is

$$\begin{aligned} &E_{i,n}^{\mathrm{B}}(t+1) = \\ &[E_{i,n}^{\mathrm{B}}(t) - g_{i,n}^{\mathrm{B}}(t) \sum_{k=1}^{D_{i,n}^{\mathrm{B,t}}(t)} 10 T_{\mathrm{s}} p_{i,n}^{\mathrm{B}}(t,k)]^+ + E_{i,n}^{\mathrm{d,r}}(t), \end{aligned} \tag{18}$$

where $D_{i,n}^{\mathrm{B,t}}(t)$ is the transmission time in the present updating circle, which is a discrete in time slots. $E_{i,n}^{\mathrm{d,r}}(t)$ is the amount of solar energy harvested in the present updating circle, which is unknown at the beginning of the updating circle. If the remaining energy is larger than a working threshold, then it is available to use in the next updating circle. That is

$$v_{i,n}^{\mathrm{B}}(t+1)=\begin{cases}1, & \text{if } E_{i,n}^{\mathrm{B}}(t+1)\geq E_{\mathrm{th}}^{\mathrm{B}},\\ 0, & \text{else.}\end{cases} \tag{19}$$

The consumed energy on an eMBB sensor is dynamically changing in different updating circles. Therefore, working threshold can not be set by the energy consumption. To guarantee the energy consumption under the remaining energy on eMBB sensors, the following energy constraint should be met

$$\sum_{k=1}^{D_{i,n}^{\mathrm{B,t}}(t)} 10T_{\mathrm{s}}p_{i,n}^{\mathrm{B}}(t,k)\leq E_{i,n}^{\mathrm{B}}(t),\quad \forall i\in\mathcal{F},n\in\mathcal{D}_i^{\mathrm{B}}. \tag{20}$$

4.5 Objective

The aim of this paper is to minimize the average long-term energy of the overall DT-WA updating system, under updating convergency and latency constraint, formulated by

$$\begin{aligned}\min_{\{\mathbf{m},\mathbf{f},\mathbf{P}^{\mathrm{B,r}},\mathbf{g}^{\mathrm{B}},\mathbf{g}^{\mathrm{U}},\mathbf{I}\}} & \lim_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T}\sum_{i=1}^{F}E_i(t),\\ \text{s.t. } & C1: g_{i,n}^{\mathrm{B}}(t)\leq v_{i,n}^{\mathrm{B}}(t),\quad \forall i\in\mathcal{F},n\in\mathcal{D}_i^{\mathrm{B}},\\ & C2: g_{i,n}^{\mathrm{U}}(t)\leq v_{i,n}^{\mathrm{U}}(t),\quad \forall i\in\mathcal{F},n\in\mathcal{D}_i^{\mathrm{U}},\\ & C3: \sum_{k=1}^{D_{i,n}^{\mathrm{B,t}}(t)} 10T_{\mathrm{s}}p_{i,n}^{\mathrm{B}}(t,k)\leq E_{i,n}^{\mathrm{B}}(t),\\ & C4: \varepsilon_i(t)\geq\varepsilon_{th},\quad \forall t\in\{1,\cdots,\infty\},\\ & C5: D_i(t)\leq D_{\mathrm{th}},\quad \forall t\in\{1,\cdots,\infty\},\end{aligned} \tag{21}$$

where $\mathbf{m}=\{m_i(t)|i\in\mathcal{F}\}$, $\mathbf{f}=\{f_i(t)|i\in\mathcal{F}\}$, $\mathbf{P}^{\mathrm{B,r}}=\{P_i^{\mathrm{B,r}}(t)|i\in\mathcal{F}\}$, $\mathbf{g}^{\mathrm{B}}=\{g_{i,n}^{\mathrm{B}}(t)|n\in\mathcal{D}_i^{\mathrm{B}},i\in\mathcal{F}\}$, $\mathbf{g}^{\mathrm{U}}=\{g_{i,n}^{\mathrm{U}}(t)|n\in\mathcal{D}_i^{\mathrm{U}},i\in\mathcal{F}\}$, $\mathbf{I}=\{I_i(t)|i\in\mathcal{F}\}$. $C1$ and $C2$ express that only those sensors with abundant energy are available to be activated. $C3$ limits the eMBB sensors energy consumption under their remaining energy. $C4$ and $C5$ limit the updating convergency and latency, which are critical metrics in AI applications [22].

From Eq. (9) and Eq. (13), we can see that more activated sensors would lead to a less computing workload under given convergency requirement. This insight indicates that ESs with more available sensors are more likely to be the helpers to reduce its surrounding energy. However, the cooperation process requires helpers to reduce their computing latency, so as to leave more time for their recipients to further compute. It would increase the computing energy of helper, inducing energy sacrifice.

V. COOPERATION AND RESOURCE ALLOCATION ALGORITHM

5.1 Overview of Whole Algorithm

We employ multi-agent deep reinforcement learning (MA-DRL) to facilitate the ESs decision making process with the aim to minimize the system energy. Each ES deploys a DRL agent which dynamically generate decisions (e.g., allocate resources) based on local environment (e.g., number of available sensors) in the beginning of each updating circle, to collaboratively update sub-models for DT-WA in the dynamic environment. Each agent would experience two phases, including offline training and online implementing. In the offline training phase, each ES trains its agent's actor network and critic network based on the interacting performance with physical environment. In the online implementing phase, each ES uses its trained actor network to generate actions.

To reach the goal, first, we reformulates the original problem as a Dec-POMDP. Then, the dimensions of action and state vectors are reduced, where a child optimization problem is solved to minimize the local computing energy of each ES. After that, several constraint-released methods are designed to meet the sub-model updating convergency and latency requirements, where another child optimization problem is solved to minimize the surrounding energies of helper ESs. Moreover, we introduce the mean-field theory to propose a distributed large-scale MA-DRL which is feasible for the large-scale agent scenario. Finally, we give the overall algorithm and computing complexity.

5.2 Dec-POMDP Formulation

The original problem can be reformulated into a Dec-POMDP described by a tuple $\langle S,A,P,R,\mathbb{O},\gamma\rangle$, where each element is given as follows:

5.2.1 State Space

In the distributed DT-WA model updating process, the global state space is defined as $S = \{S_i | i \in \mathcal{F}\}$, where S_i is the local state space of ES i given by Eq. (22).

$$S_i = \Big(s_i = \Big(\{v_{i,n}^{\mathrm{B}}\}_{n\in\mathcal{D}_i^{\mathrm{B}}}, \{v_{i,n}^{\mathrm{U}}\}_{n\in\mathcal{D}_i^{\mathrm{B}}}, \{E_{i,n}^{\mathrm{B}}\}_{n\in\mathcal{D}_i^{\mathrm{B}}}, \{E_{i,n}^{\mathrm{U}}\}_{n\in\mathcal{D}_i^{\mathrm{B}}}, u_i^{\mathrm{base}}\Big) | v_{i,n}^{\mathrm{B}} \in \{0,1\}, v_{i,n}^{\mathrm{U}} \in \{0,1\}, u_i^{\mathrm{base}} \in [0,1], E_{i,n}^{\mathrm{B}} \in (0, E_{\max}^{\mathrm{B}}), E_{i,n}^{\mathrm{U}} \in (0, E_{\max}^{\mathrm{U}})\Big). \tag{22}$$

In the Eq. (22), $v_{i,n}^{\mathrm{B}}$ and $v_{i,n}^{\mathrm{U}}$ are the availability variables of eMBB sensors and URLLC sensors under ES i, $E_{i,n}^{\mathrm{B}}$ and $E_{i,n}^{\mathrm{U}}$ are the sensor remaining energy, u_i^{base} is the basic convergency factor.

5.2.2 Action Space

The global action space is defined as $A = \{A_i | i \in \mathcal{F}\}$, where A_i is the local action space of ES i given by Eq. (23).

$$A_i = (a_i = \Big(m_i, \{g_{i,n}^{\mathrm{B}}\}_{n\in\mathcal{D}_i^{\mathrm{B}}}, \{g_{i,n}^{\mathrm{U}}\}_{n\in\mathcal{D}_i^{\mathrm{B}}}, p_i^{\mathrm{B,r}}, f_i, I_i^{\mathrm{B}}, I_i^{\mathrm{U}}, I_i^{\mathrm{Int}}\Big) | m_i \in \{0,1\}, g_{i,n}^{\mathrm{B}} \in \{0,1\}, g_{i,n}^{\mathrm{U}} \in \{0,1\}, p_i^{\mathrm{B,r}} \in [P_{\min}^{\mathrm{B,r}}, P_{\max}^{\mathrm{B,r}}], f_i \in [0, f_{\max}], I_i^{\mathrm{B}} \in \{0,1,\cdots,\infty\}, I_i^{\mathrm{U}} \in \{0,1,\cdots,\infty\}, I_i^{\mathrm{Int}} \in \{0,1,\cdots,\infty\}, \Big). \tag{23}$$

In the Eq. (23), m_i is the cooperative identity chosen by ES i, $g_{i,n}^{\mathrm{B}}$ and $g_{i,n}^{\mathrm{U}}$ are the activating variables of eMBB sensors and URLLC sensors under ES i, $p_i^{\mathrm{B,r}}$ is the received power at BS connected with ES i, f_i is the amount of allocated computing resources at ES i, $I_i^{\mathrm{B}}, I_i^{\mathrm{U}}, I_i^{\mathrm{Int}}$ are the training iteration time for eMBB and URLLC data-related child networks and the integrated network in the sub-model on ES i, respectively. The actions needs to meet system constrains in $C1 - C5$.

5.2.3 System State Transition Probability

The transition probability of global state is related with the transition probability of local states:

$$P(s(t+1)|s(t), a(t)) = \prod_{i\in\mathcal{F}} P(s_i(t+1)|s_i(t), a_i(t)), \tag{24}$$

where the transition probability of local states is related with the local actions and local observations by Eq. (25). From the Eq. (25), the transition probability is unknown due to the random solar energy harvesting on sensors and randomly changing environment.

$$\begin{aligned} &P(s_i(t+1)|s_i(t), a_i(t)) \\ &= \prod_{n\in\mathcal{D}_i^{\mathrm{B}}} \underbrace{P\left(v_{i,n}^{\mathrm{B}}(t+1)|v_{i,n}^{\mathrm{B}}(t), E_{i,n}^{\mathrm{B}}(t), g_{i,n}^{\mathrm{B}}(t), p_i^{\mathrm{B,r}}(t)\right)}_{\text{See Eq. (18) and (19) with unknown harvested energy.}} \times \prod_{n\in\mathcal{D}_i^{\mathrm{U}}} \underbrace{P\left(v_{i,n}^{\mathrm{U}}(t+1)|v_{i,n}^{\mathrm{U}}(t), E_{i,n}^{\mathrm{U}}(t), g_{i,n}^{\mathrm{U}}(t)\right)}_{\text{See (16) and (17) with unknown harvested energy.}} \\ &\times \prod_{n\in\mathcal{D}_i^{\mathrm{B}}} \underbrace{P\left(E_{i,n}^{\mathrm{B}}(t+1)|E_{i,n}^{\mathrm{B}}(t), g_{i,n}^{\mathrm{B}}(t), p_i^{\mathrm{B,r}}(t)\right)}_{\text{See (18) with unknown harvested energy.}} \times \prod_{n\in\mathcal{D}_i^{\mathrm{U}}} \underbrace{P\left(E_{i,n}^{\mathrm{U}}(t+1)|E_{i,n}^{\mathrm{U}}(t), g_{i,n}^{\mathrm{U}}(t)\right)}_{\text{See (16)) with unknown harvested energy.}} \\ &\times \underbrace{P\left(u_i^{\mathrm{base}}(t+1)|u_i^{\mathrm{base}}(t), a_i(t)\right)}_{\text{Randomly related with the current convergency (13).}}. \end{aligned} \tag{25}$$

5.2.4 Reward Function

Each ES can only know local and neighboring observations, based on which it learns to minimize the long-term average system energy. Considering each ES's decision mainly influence itself and neighboring energy consumption, and the local reward function can only be designed based on the neighboring information, we design the local reward function to be

$$
\begin{aligned}
&R_i(s_i(t), \{s_j(t)\}_{j\in\mathcal{B}_i}, a_i(t), \{a_j(t)\}_{j\in\mathcal{B}_i}) \\
&= -E_i(t) - \sum_{j\in\mathcal{B}_i} E_j(t),
\end{aligned} \tag{26}
$$

which considers both the local and neighboring energy consumption. It reflects the main energy impact of each ES's decision on system energy, and thus can lead the ESs to jointly achieve an approximate minimized overall energy.

5.2.5 Observations

The joint observation space is a set of local observation space $\mathbb{O} = \{\mathbb{O}_i | i \in \mathcal{F}\}$, the latter can be defined to be the same as the local state space S_i. Similarly, observation vector o_i is defined to be the same as local state vector s_i.

5.2.6 Discount Factor

The goal of each agent is to learn policies to maximize the long-term discounted accumulated reward. Therefore, we set the discount factor γ as 0.99 to make the agent's goal consistent with aim to minimize the long-term system energy.

5.3 Transformation in Observation and Action

We use two approaches to reduce the dimensions of the instant local observation vector $o_i(t)$ and action vector $a_i(t)$, which could facilitate the latter learning process in the MA-DRL algorithm.

5.3.1 Replacing Large Sensor-Related Vectors into Single Elements

The dimension of instant local observation and action is unacceptably large. The main reason is that the dimension of sensor availability vectors $\left\{v_{i,n}^{\mathrm{B}}(t)\right\}_{n\in\mathcal{D}_i^{\mathrm{B}}}$, $\left\{v_{i,n}^{\mathrm{U}}(t)\right\}_{n\in\mathcal{D}_i^{\mathrm{U}}}$, and the remaining battery vector $\left\{E_{i,n}^{\mathrm{B}}(t)\right\}_{n\in\mathcal{D}_i^{\mathrm{B}}}$, $\left\{E_{i,n}^{\mathrm{U}}(t)\right\}_{n\in\mathcal{D}_i^{\mathrm{U}}}$ are equal to the number of heterogenous sensors under ES i, which can be very large. To cope with the problem, we replace the sensor availability vectors with the number of available heterogeneous sensors $N_i^{\mathrm{B,v}}(t) = \sum_{n=1}^{N_i^{\mathrm{B}}} v_{i,n}^{\mathrm{B}}(t)$ and $N_i^{\mathrm{U,v}}(t) = \sum_{n=1}^{N_i^{\mathrm{U}}} v_{i,n}^{\mathrm{U}}(t)$. Then, we sort the available sensors according to their remaining energies and channel gains, and choose the first $N_i^{\mathrm{B,a}}(t)$ and $N_i^{\mathrm{U,a}}(t)$ sensors to activate. This process can delete the remaining energy vectors from action vector, and replace the large-dimensional activating vectors $\left\{g_{i,n}^{\mathrm{B}}(t)\right\}_{n\in\mathcal{D}_i^{\mathrm{B}}}$, $\left\{g_{i,n}^{\mathrm{U}}(t)\right\}_{n\in\mathcal{D}_i^{\mathrm{U}}}$ by $N_i^{\mathrm{B,a}}(t), N_i^{\mathrm{U,a}}(t)$. The constraints in this process include the activation constraint and energy constraint on sensors:

$$
N_i^{\mathrm{B,a}}(t) \le N_i^{\mathrm{B,v}}(t),\ N_i^{\mathrm{U,a}}(t) \le N_i^{\mathrm{U,v}}(t),\ C3. \tag{27}
$$

5.3.2 Replacing Iteration Number with Convergency Variable

In action vector, the training iteration times $I_i^{\mathrm{B}}(t)$, $I_i^{\mathrm{U}}(t)$ and $I_i^{\mathrm{Int}}(t)$ have continues-like action spaces, which is extremely large. Moreover, it influences the reward through complex relationships in energy consumptions and updating convergencies, which makes it difficult for agents to learn to generate the optimal iteration number. To cope with the problem, we rewrite the constraint $C3$ into (28).

$$
\begin{aligned}
&\underbrace{u_i^{\mathrm{base}}(t)}_{z_i^{\mathrm{base}}(t)\cdot A} + \underbrace{(k^{\mathrm{B}} I_i^{\mathrm{B}}(t)\cdot[Q_i^{\mathrm{B}}(t)]^v + k^{\mathrm{U}} I_i^{\mathrm{U}}(t)\cdot[Q_i^{\mathrm{U}}(t)]^v + k^{\mathrm{Int}} I_i^{\mathrm{Int}}(t)\cdot[\min\{Q_i^{\mathrm{B}}(t), Q_i^{\mathrm{U}}(t)\}]^v)}_{z_i(t)\cdot A} \\
&= (z_i^{\mathrm{base}}(t) + z_i(t))A \ge A.
\end{aligned} \tag{28}
$$

In (28), $A = -\ln(1-\varepsilon_{th})$. In (28), we introduce two new variables $z_i^{\mathrm{base}}(t)$ and $z_i(t)$, which are named as (new) basic convergency factor and convergency variable, where

$$
z_i(t) \ge 1 - z_i^{\mathrm{base}}(t). \tag{29}
$$

Equation (28) gives the relationships between the convergency variable with the training iteration times $I_i^{\mathrm{B}}(t)$, $I_i^{\mathrm{U}}(t)$ and $I_i^{\mathrm{Int}}(t)$, and the dataset dimensions $Q_i^{\mathrm{B}}(t)$. Next, we want to transform the action vector by (30).

$$a_i(t) = \left(N_i^{\mathrm{B,a}}(t), N_i^{\mathrm{U,a}}(t), I_i^{\mathrm{B}}(t), I_i^{\mathrm{U}}(t), I_i^{\mathrm{Int}}(t), \cdots\right) \quad \Rightarrow \left(N_i^{\mathrm{B,a}}(t), N_i^{\mathrm{U,a}}(t), z_i(t), \cdots\right). \tag{30}$$

In (30), the training iteration times $I_i^{\mathrm{B}}(t)$, $I_i^{\mathrm{U}}(t)$ and $I_i^{\mathrm{Int}}(t)$ become dependent variables of the convergency variable $z_i(t)$ and sensor activated numbers $N_i^{\mathrm{B,a}}(t)$, $N_i^{\mathrm{U,a}}(t)$.

To do this, we formulate a problem to minimize the local computing energy in (31).

$$\begin{aligned}\min_{I_i^{\mathrm{B}}(t), I_i^{\mathrm{U}}(t), I_i^{\mathrm{Int}}(t)} \quad & I_i^{\mathrm{B}}(t) \cdot Q_i^{\mathrm{B}}(t) \cdot c^{\mathrm{B}} + I_i^{\mathrm{U}}(t) \cdot Q_i^{\mathrm{U}}(t) \cdot c^{\mathrm{U}} + I_i^{\mathrm{Int}}(t) \cdot \min\{Q_i^{\mathrm{B}}(t), Q_i^{\mathrm{U}}(t)\} \cdot c^{\mathrm{Int}}, \\ \text{s.t. } & k^{\mathrm{U}} I_i^{\mathrm{U}}(t) \cdot [Q_i^{\mathrm{U}}(t)]^v + k^{\mathrm{B}} I_i^{\mathrm{B}}(t) \cdot [Q_i^{\mathrm{B}}(t)]^v + k^{\mathrm{Int}} I_i^{\mathrm{Int}}(t) \cdot \min\{Q_i^{\mathrm{B}}(t), Q_i^{\mathrm{U}}(t) = z_i(t) \cdot A.\end{aligned} \tag{31}$$

It is to find the optimal iteration times under given sensor numbers $N_i^{\mathrm{B,a}}(t)$, $N_i^{\mathrm{U,a}}(t)$ and convergency factor $z_i(t)$, where $Q^{\mathrm{B}} = l^{\mathrm{B}} N_i^{\mathrm{B,a}}(t)$, $Q^{\mathrm{U}} = l^{\mathrm{U}} N_i^{\mathrm{U,a}}(t)$ in (31). It can be easily transformed into (32) by combining the equation in (31) into the minimization problem.

$$\begin{aligned}&\min_{I_i^{\mathrm{B}}(t), I_i^{\mathrm{U}}(t)} \left[I_i^{\mathrm{B}}(t) \cdot Q_i^{\mathrm{B}}(t) c^{\mathrm{B}} + I_i^{\mathrm{U}}(t) Q_i^{\mathrm{U}}(t) c^{\mathrm{U}} + \underbrace{\frac{z_i(t) \cdot A - (k^{\mathrm{U}} I_i^{\mathrm{U}}(t) \cdot [Q_i^{\mathrm{U}}(t)]^v + k^{\mathrm{B}} I_i^{\mathrm{B}}(t) \cdot [Q_i^{\mathrm{B}}(t)]^v)}{k^{\mathrm{Int}} \cdot \min\{Q_i^{\mathrm{B}}(t), Q_i^{\mathrm{U}}(t)\}^{v-1}}}_{I_i^{\mathrm{Int}}(t) \cdot \min\{Q_i^{\mathrm{B}}(t), Q_i^{\mathrm{U}}(t)\}} \cdot c^{\mathrm{Int}} \right] \\ &= \min_{I_i^{\mathrm{B}}(t), I_i^{\mathrm{U}}(t)} \left[\frac{z_i(t) \cdot A \cdot c^{\mathrm{Int}}}{k^{\mathrm{Int}} \min\{Q_i^{\mathrm{B}}(t), Q_i^{\mathrm{U}}(t)\}^{v-1}} + I_i^{\mathrm{B}}(t) Q_i^{\mathrm{B}}(t) \underbrace{\left(\frac{k^{\mathrm{Int}} c^{\mathrm{B}} \min\{Q_i^{\mathrm{B}}(t), Q_i^{\mathrm{U}}(t)\}^{v-1} - k^{\mathrm{B}} c^{\mathrm{Int}} Q_i^{\mathrm{B}}(t)^{v-1}}{k^{\mathrm{Int}} \cdot \min\{Q_i^{\mathrm{B}}(t), Q_i^{\mathrm{U}}(t)\}^{v-1}} \right)}_{①} \right. \\ &\qquad \left. + I_i^{\mathrm{U}}(t) \cdot Q_i^{\mathrm{U}}(t) \underbrace{\left(\frac{k^{\mathrm{Int}} c^{\mathrm{U}} \cdot \min\{Q_i^{\mathrm{B}}(t), Q_i^{\mathrm{U}}(t)\}^{v-1} - k^{\mathrm{U}} c^{\mathrm{Int}} Q_i^{\mathrm{U}}(t)^{v-1}}{k^{\mathrm{Int}} \cdot \min\{Q_i^{\mathrm{B}}(t), Q_i^{\mathrm{U}}(t)\}^{v-1}} \right)}_{②} \right].\end{aligned} \tag{32}$$

Then, the optimal iterations can be derived as dependent variables of dataset dimensions and convergency variable, which is given in (33).

$$\begin{cases} I_i^{\mathrm{B}}(t)^* = 0, I_i^{\mathrm{U}}(t)^* = 0, I_i^{\mathrm{Int}}(t)^* = \frac{z_i(t) \cdot A}{k^{\mathrm{Int}} \cdot \min\{Q_i^{\mathrm{B}}(t), Q_i^{\mathrm{U}}(t)\}^v}, \text{ if } ① \geq 0, ② \geq 0, \\ I_i^{\mathrm{B}}(t)^* = 0, I_i^{\mathrm{U}}(t)^* = \frac{z_i(t) \cdot A}{k^{\mathrm{U}} \cdot [Q_i^{\mathrm{U}}(t)]^v}, I_i^{\mathrm{Int}}(t)^* = 0, \text{ if } ② < 0, \frac{c^{\mathrm{U}} k^{\mathrm{B}} Q_i^{\mathrm{B}}(t)^v}{k^{\mathrm{U}} Q_i^{\mathrm{U}}(t)^{v-1}} \leq c^{\mathrm{B}} Q_i^{\mathrm{B}}(t), \\ I_i^{\mathrm{B}}(t)^* = \frac{z_i(t) \cdot A}{k^{\mathrm{B}} \cdot [Q_i^{\mathrm{B}}(t)]^v}, I_i^{\mathrm{U}}(t)^* = 0, I_i^{\mathrm{Int}}(t)^* = 0, \text{ if } ① < 0, \frac{c^{\mathrm{U}} k^{\mathrm{B}} Q_i^{\mathrm{B}}(t)^v}{k^{\mathrm{U}} Q_i^{\mathrm{U}}(t)^{v-1}} > c^{\mathrm{B}} Q_i^{\mathrm{B}}(t). \end{cases} \tag{33}$$

The corresponding workload is given in (34).

$$\begin{cases} C_i(t) = \dfrac{z_i(t)\cdot A\cdot c^{\mathrm{Int}}}{k_i^{\mathrm{Int}}\cdot\min\{Q_i^{\mathrm{B}}(t),Q_i^{\mathrm{U}}(t)\}^{v-1}}, & \text{if ① } \geq 0, \text{ ② } \geq 0, \\ C_i(t) = \dfrac{z_i(t)\cdot A\cdot c^{\mathrm{U}}}{k_i^{\mathrm{U}}\cdot[Q_i^{\mathrm{U}}(t)]^{v-1}}, & \text{if ① } \geq 0, \text{ ② } < 0, \text{ or ① } < 0, \text{ ② } < 0, \ \frac{c^{\mathrm{U}}k_i^{\mathrm{B}}Q_i^{\mathrm{B}}(t)^v}{k_i^{\mathrm{U}}Q_i^{\mathrm{U}}(t)^{v-1}} \leq c^{\mathrm{B}}Q_i^{\mathrm{B}}(t), \\ C_i(t) = \dfrac{z_i(t)\cdot A\cdot c^{\mathrm{B}}}{k_i^{\mathrm{B}}\cdot[Q_i^{\mathrm{B}}(t)]^{v-1}}, & \text{if ① } < 0, \text{ ② } \geq 0, \text{ or ① } < 0, \text{ ② } < 0, \ \frac{c^{\mathrm{U}}k_i^{\mathrm{B}}Q_i^{\mathrm{B}}(t)^v}{k_i^{\mathrm{U}}Q_i^{\mathrm{U}}(t)^{v-1}} > c^{\mathrm{B}}Q_i^{\mathrm{B}}(t). \end{cases} \quad (34)$$

The equation gives us an insight that if the available number of one type of sensor is obviously larger than the other one (e.g., $Q_i^{\mathrm{B}}(t) > Q_i^{\mathrm{U}}(t)$ and $\frac{k^{\mathrm{Int}}c^{\mathrm{B}}}{k^{\mathrm{B}}c^{\mathrm{Int}}} < \frac{Q_i^{\mathrm{B}}(t)}{Q_i^{\mathrm{U}}(t)}$), it is better to only activate the former type of sensor, to meet the convergency requirement while saving the latter sensor remaining energy for future updating circles.

Therefore, the modified observation and action vectors are

$$\begin{aligned} & o_i(t) = \left(N_i^{\mathrm{B,v}}(t), N_i^{\mathrm{U,v}}(t), z_i^{\mathrm{base}}(t)\right), i \in \mathcal{F}, \\ & a_i(t) = \left(m_i(t), N_i^{\mathrm{B,a}}(t), N_i^{\mathrm{U,a}}(t), p_i^{\mathrm{B,r}}(t), f_i(t), z_i(t)\right), \\ & \text{s.t. } (27), (29), C5. \end{aligned} \quad (35)$$

5.4 Constraint-Released Methods

We employ different constraint-released methods to meet different constraints in (35) to change the constraint-MG into MG.

5.4.1 Latency Constraint

The "surrounding energy" refers to total computing energy of each helper and its recipients. To meet latency constraint, helpers need to determine the computing resource allocation for itself and its recipients. The detail is given as follows.

Latency and energy are negatively correlated. The updating time of each helper is part of the updating time of their recipients. Take the helper i and its recipient set $\mathcal{G}_i$ for example. We set the total updating latency of recipients equal to the latency threshold. Then, we formulate a child optimization problem to minimize the helper and its recipients energy $E_i^{\mathrm{surr}}(t) = E_i(t) + \sum_{j\in\mathcal{G}_i} E_j(t)$ by finding the optimal computing latency $D_i^{\mathrm{c}}(t)$ of helper i in (36).

$$\begin{aligned} & \min_{D_i^{\mathrm{c}}(t)} E_i^{\mathrm{surr}}(t) = \min_{D_i^{\mathrm{c}}(t)} \underbrace{\frac{\kappa\cdot[C_i(t)]^3}{[D_i^{\mathrm{c}}(t)]^2}}_{E_i(t)} + \sum_{j\in\mathcal{G}_i} \underbrace{\frac{\kappa\cdot[C_j(t)]^3}{[D_{\mathrm{th}} - D_i^{\mathrm{t}}(t) - D_i^{\mathrm{c}}(t)]^2}}_{E_j(t)}, \\ & \text{s.t.,} \ \ D_i^{\mathrm{c}}(t) + D_i^{\mathrm{t}}(t) < D_{\mathrm{th}}. \end{aligned} \quad (36)$$

Its derivative with respect to $D_i^{\mathrm{c}}(t)$ is given in (37).

$$E_i^{\mathrm{surr}}(t)' = 2\kappa\cdot\frac{-[C_i(t)]^3\cdot[D_{\mathrm{th}} - D_i^{\mathrm{t}}(t) - D_i^{\mathrm{c}}(t)]^3 + [D_i^{\mathrm{c}}(t)]^3\cdot\sum_{j\in\mathcal{G}_i}[C_j(t)]^3}{[D_i^{\mathrm{c}}(t)]^3\cdot[D_{\mathrm{th}} - D_i^{\mathrm{t}}(t) - D_i^{\mathrm{c}}(t)]^3}. \quad (37)$$

In (37), the denominator is always positive because of the latency constraint in (36). Therefore, the sign of the derivative is determined by the numerator which is strictly increasing with the increase of computing latency $D_i^{\mathrm{c}}(t)$. Moreover, the numerator becomes zero when the $D_i^{\mathrm{c}}(t)$ is equal to the value in (38), which indicates the optimizing objective in (36) is a concave function, and (38) gives the optimal computing time of helper i and its recipients.

$$\begin{aligned} D_i^{\mathrm{c}}(t)^* &= \frac{C_i(t)\cdot(D_{\mathrm{th}}-D_i^{\mathrm{t}}(t))}{C_i(t)+[\sum_{j\in\mathcal{G}_i} C_j^3(t)]^{1/3}}, \\ D_j^{\mathrm{c}}(t)^* &= D_{\mathrm{th}} - D_i^{\mathrm{t}}(t) - D_i^{\mathrm{c}}(t)^*, \quad j\in\mathcal{G}_i. \end{aligned} \tag{38}$$

Given the optimal computing time, the optimal computation resource is $f_i(t) = C_i(t)/D_i^{\mathrm{c}}(t)^*$. Thus, the computation resource is reduced from the action vector by

$$\begin{aligned} a_i(t) &= \left(m_i(t), N_i^{\mathrm{B,a}}(t), N_i^{\mathrm{U,a}}(t), p_i^{\mathrm{B,r}}(t), f_i(t), z_i(t)\right) \\ &\Rightarrow \left(m_i(t), N_i^{\mathrm{B,a}}(t), N_i^{\mathrm{U,a}}(t), p_i^{\mathrm{B,r}}(t), z_i(t)\right). \end{aligned} \tag{39}$$

5.4.2 Accuracy Constraint

For a helper, if its convergency action variable do not meet constraint, a punishment is multiplied with the reward function

$$r_i(t) = -(E_i(t) + E_{\mathcal{B}_i}(t))\cdot p_i^{\mathrm{a}}(t), \tag{40}$$

where $p_i^{\mathrm{a}}(t) = \left(1+\left[\frac{1-z_i^{\mathrm{base}}(t)-z_i(t)}{1-z_i^{\mathrm{base}}(t)}\right]^+\right)$. For a recipient, its sub-model's basic updating convergency is modified when receiving helper's sub-model parameters. Then, if the recipient's convergency variable does not meet the constraint, it is rectified to meet the requirement by

$$z_j(t) = \min\{z_j(t), 1-z_j^{\mathrm{base}}(t)\}. \tag{41}$$

The computing resource needs to recalculated based on the rectified convergency variable and the optimal latency in (38).

5.4.3 Sensor's Constraint

Constraints in (27) includes sensors activating constraints and energy constraints. The activating constraints limit the sensor activating number under their available number. To deal with this constraint, we let the agent actor network generate normalized activating number, and scale it with the number of available sensors. The energy constraints convey that the energy consumption of sensors cannot surpass its remaining battery energy. The energy constraint of URLLC sensor has been met in its activating constraint.

5.5 Distributed Large-Scale MA-DRL

Multi-agent deep reinforcement learning (MA-DRL) has been proved as an effective approach to solve Dec-POMDP [23]. In particular, multi-agent proximal policy optimization (MA-PPO) [23] has emerged as an outstanding algorithm to speed up the training convergence and obtain high rewards. However, it only work well under a small number of agents $(2 \sim 4)$ because of the dimensional curse, and relies on centralized training process which can cause heavy communication cost.

To tackle the dimensional curse, we leverage the mean-field theory to modify the value function (critic network) and actor function (actor network) in MA-PPO. Specifically, the mean-field theory conveys that the joint influences from other agents on one agent may be approximated as the mean influence from neighboring agents in a massive agents scenario. Inspired by this theory, first, we found that the value function $V_i^{\phi}(s)$ [23] in MA-PPO can be reduced into a dependent variable of local observation $o_i(t)$ and mean neighboring observation in our content, which could dramatically reduce its input dimension, as proved by Eq. (42).

$$\begin{aligned} V_i^{\phi}(s) &= \frac{1}{|\mathcal{B}_i|}\sum_{j\in\mathcal{B}_i} V_i^{\phi}(o_i,o_j) = \frac{1}{|\mathcal{B}_i|}\sum_{j\in\mathcal{B}_i}\left[V_i^{\phi}(o_i,\bar{o}_{i-}) + \nabla_{\bar{o}_{i-}}V_i^{\phi}(o_i,\bar{o}_{i-})\delta o_{i,j} + \frac{1}{2}\delta o_{i,j}\nabla^2_{\tilde{s}_{i,j}}V_i^{\phi}(o_i,\tilde{s}_{i,j})\delta o_{i,j}\right] \\ &= V_i^{\phi}(o_i,\bar{o}_{i-}) + \nabla_{\bar{o}_{i-}}V_i^{\phi}(o_i,\bar{o}_{i-})\cdot\left[\frac{1}{|\mathcal{B}_i|}\sum_{j\in\mathcal{B}_i}\delta o_{i,j}\right] + \frac{1}{2|\mathcal{B}_i|}\sum_{j\in\mathcal{B}_i}\left[\delta o_{i,j}\cdot\nabla^2_{\tilde{s}_{i,j}}V_i^{\phi}(o_i,\tilde{s}_{i,j})\cdot\delta o_{i,j}\right] \\ &= V_i^{\phi}(o_i,\bar{o}_{i-}) + \frac{1}{2|\mathcal{B}_i|}\sum_{j\in\mathcal{B}_i} T_i(o_j) \approx V_i^{\phi}(o_i,\bar{o}_{i-}). \end{aligned} \tag{42}$$

The first equality in (42) satisfies because the local reward $r_i(t)$ can be approximated to only dependent on local observation $o_i(t)$ and neighboring observations $o_j(t)$ $(j \in \mathcal{B}_i)$, according to (26). $T_i(o_j) \triangleq \left[\delta o_{i,j} \cdot \nabla^2_{\tilde{o}_{i,j}} V_i^{\phi}(o_i, \tilde{o}_{i,j}) \cdot \delta o_{i,j}\right]$ is the Taylor polynomial's remainder with $\tilde{s}_{i,j} = \overline{o}_{i-} + \epsilon_{i,j}\delta o_{i,j}$ and $\epsilon_{i,j} \in [0,1]$. $T_i(o_j)$ is bounded within a symmetric interval $[-2M, 2M]$ when $V_i^{\phi}(o_i, o_j)$ is M-smooth [24].

Second, we slightly increase the input dimension of actor function $\pi_i^{\theta}(o_i)$ in MA-PPO to make it easier to learn the optimal decision. Specifically, we provide it with more information through its input, adding the mean neighboring observation as another independent variable in the actor function, i.e., $\pi_i^{\theta}(o_i) \Rightarrow \pi_i^{\theta}(o_i, \overline{o}_{i-})$. The effectiveness of the modifications is proved in the simulation part by comparing the proposed algorithm with the traditional MA-PPO algorithm.

5.6 Overall of Proposed Algorithm

Each ES deploys a PPO agent according to the proposed distributed large-scale MA-PPO algorithm. We change all of the action values into a (10×1) one-hot form (except for the binary variable $m_i(t)$), as a ratio to the maximum action values, from $0/10$ to $10/10$. Each agent has a critic network and actor network, both of them are in fully-connected structure. The dimensions of the input, hidden and output layer in a critic network are $L_{\mathrm{I}}^{\mathrm{C}} = 4$, $L_{\mathrm{H}}^{\mathrm{C}} \times 2 = 128 \times 2$, and 1, respectively, whose dimensions in an actor network are $L_{\mathrm{I}}^{\mathrm{A}} = 4, L_{\mathrm{H}}^{\mathrm{A}} \times 2 = 128 \times 2, L_{\mathrm{O}}^{\mathrm{A}} = 32$, respectively. For the activated functions, the output layer of critic network is activated by the Linear function, and actor network is Softmax function. Their hidden layers are activated by ReLU function. In addition, each agent would experience offline training and online implementing. The offline training phase is presented in Algorithm 1. First, to reduce the action and state space, we replace the large-dimensional variables of sensor related elements. Meanwhile, the training iteration variables are reduced by solving the neighboring energy minimization problem. Then, the algorithm alternates between sampling and training. During the sampling stage, each agent interacts with the environment, and uses several constraint-released methods to meet the DT-WA updating requirements. The training stage utilizes the sampling dataset to update the actor and critic network on each ES. In the online implementing phase, each ES would use its trained actor network to generate actions.

Algorithm 1. *Offline training phase of cooperation and resource allocation.*

```
1:  Initialization: For each ES, initialize the critic and actor network, heterogeneous sensors number, remaining energy vector and basic updating convergencies.
2:  /* Reducing state and action spaces */
3:  Modify the sensor related vector according to 5.3.1 and replace training iteration variables according to (33).
4:  for $step = 1, 2, ..., step_{\max}$ do
5:      /* Sampling */
6:      for $i = 1, 2, ..., L^{\text{batch}}$ do
7:          For each ES $i$, initialize an empty dataset $\mathcal{B}_i$.
8:          for $t = 1, 2, ..., L^{\text{step}}$ do
9:              Each ES $i$ uses $\pi_i(o_i, \bar{o}_i)$ to determine $m_i(t), N_i^{\mathrm{B,a}}(t), N_i^{\mathrm{U,a}}(t), p_i^{\mathrm{B,r}}(t), z_i(t)$.
10:             /* Constraint-Released Methods 2 and 3 */
11:             Rectify $z_i(t)$ and calculate optimal training iteration time.
12:             Each recipient ES randomly chooses a neighboring helper.
13:             Each helper calculates the optimal computing resource $f_i(t)$, updates its local sub-model and sends it to its recipients.
14:             /* Constraint-Released Methods 1 and 2 */
15:             Each recipient ES recalculate their optimal training iteration time and computation resource and update their local sub-models.
16:             Calculate reward for each agent.
17:         end for
18:         Calculate advantage function for each agent.
19:         Save interacting data into dataset $\mathcal{B}_i$.
20:     end for
21:     /* Training */
22:     Update $\pi_i(o_i, \bar{o}_i)$ and $\mathcal{Q}_i(o_i, \bar{o}_i)$ based on dataset $\mathcal{B}_i$.
23:     $\mathcal{B}_i \leftarrow \varnothing$.
24: end for
```

Table 1. *System parameters.*

Parameter	Value	Parameter	Value
N_0	-174 dBm/Hz	$l^{\rm B}, l^{\rm U}$	100, 10
κ	10^{-28}	$c^{\rm B}, c^{\rm U}$	1000 FLOPs
ϕ	8×10^{10}	$D_{\rm th}$	5 s
K	1000	B	10 MHz
$N_{\min}, N_{\max}$	1, 40	ε_{th}	0.9
$d_{\rm th}$	280 m	$s^{\rm B}, s^{\rm U}$	10000, 100 bits
$E_{\rm th}^{\rm B}$	1 J	q	0.001

5.7 Computation Complexity

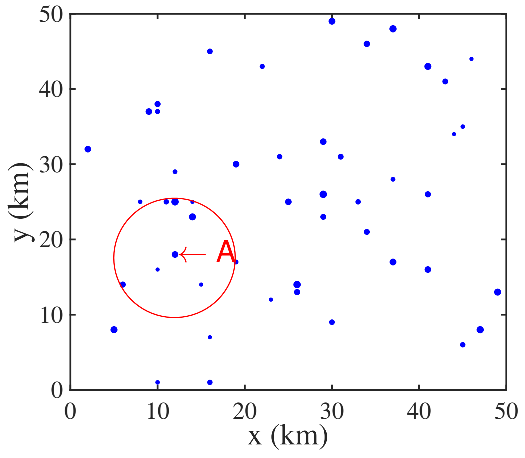
The offline training phase is shown in Algorithm 1, whose computing complexity is mainly determined by the structure of the actor network and critic network on each ES. In detail, the computing complexity of an actor network is $\mathcal{O}\left(L_{\rm I}^{\rm A} \times L_{\rm H}^{\rm A} + {L_{\rm H}^{\rm A}}^2 + L_{\rm H}^{\rm A} \times L_{\rm O}^{\rm A}\right)$. The computing complexity of a critic network is $\mathcal{O}\left(L_{\rm I}^{\rm C} \times L_{\rm H}^{\rm C} + {L_{\rm H}^{\rm C}}^2 + L_{\rm H}^{\rm C} \times 1\right)$. Therefore, the total computing complexity of Algorithm 1 is

$$\mathcal{O}\left(L_{\rm I}^{\rm A} L_{\rm H}^{\rm A} + {L_{\rm H}^{\rm A}}^2 + L_{\rm H}^{\rm A} L_{\rm O}^{\rm A} + L_{\rm I}^{\rm C} L_{\rm H}^{\rm C} + {L_{\rm H}^{\rm C}}^2 + L_{\rm H}^{\rm C}\right). \tag{43}$$

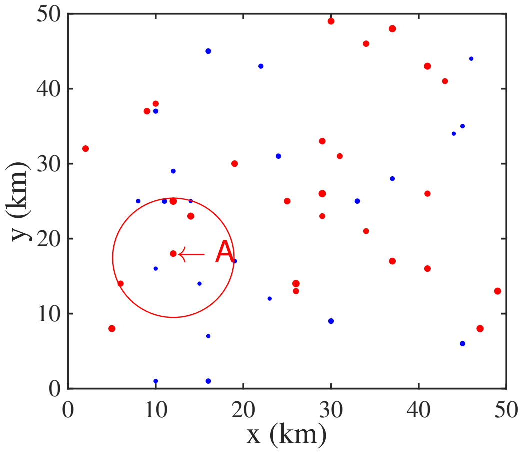
In the online implementing phase, each ES uses its trained actor network to generate decisions. Therefore, its computing complexity is equal to the computing complexity of an actor network $\mathcal{O}\left(L_{\rm I}^{\rm A} \times L_{\rm H}^{\rm A} + {L_{\rm H}^{\rm A}}^2 + L_{\rm H}^{\rm A} \times L_{\rm O}^{\rm A}\right)$. Notice that, since the algorithm has reached convergency in the offline training phase, its execution time in the implementing phase can be omitted because of the small computing complexity of the actor network.

VI. SIMULATION RESULTS AND DISCUSSIONS

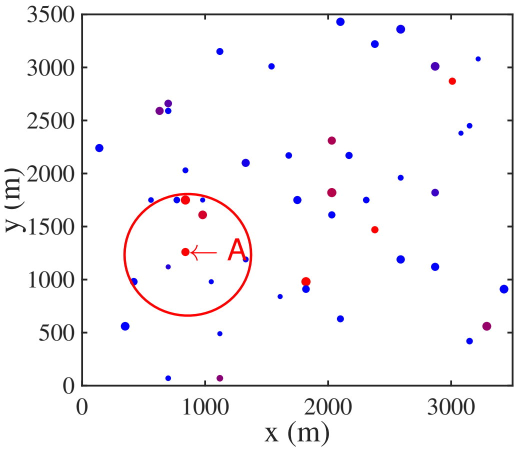
To prove the capability of the proposed algorithm under large amount of agents, we randomly deploy 50 ESs co-located with APs in 3500×3500 square meter, compared with only $2 \sim 4$ agents in traditional MA-DRL. The path loss model between ES connected AP with sensors is $38.46 + 20\log_{10}(d)$ [25]. System parameters are given in Table 1. We choose four typical baselines for comparison:



(a) *MA-PPO algorithm.*



(b) *Greedy algorithm.*



(c) *Proposed algorithm.*

Figure 2. *Mean cooperative identities under different algorithms. Each point represents an ES, whose size is proportional to its local sensors quantity, which is randomly chosen from $[N_{\min}, N_{\max}]$. The point color represents the longterm mean cooperative identity of each ES. It changes from black to red to present identity from recipient to helper.*

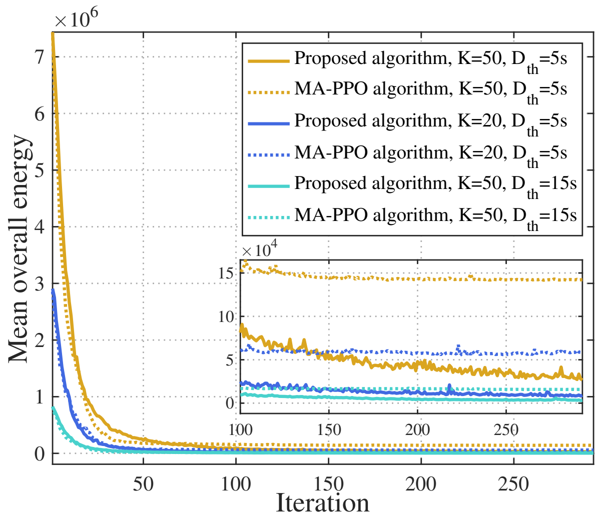


Figure 3. *Convergency performance.*

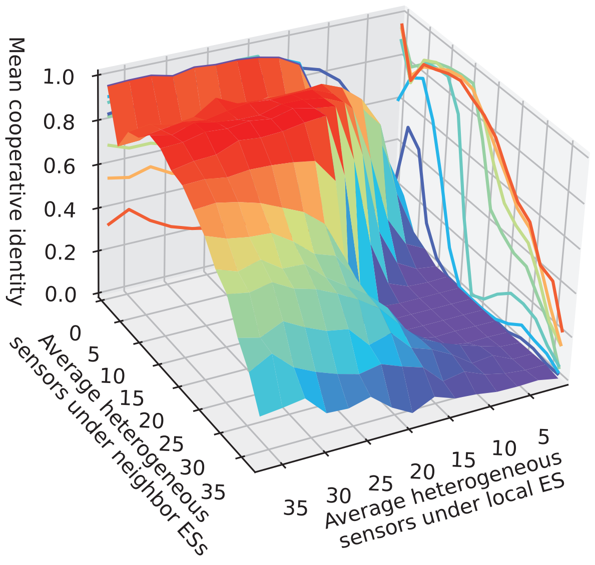


Figure 4. *Mean cooperative identity vs. Different number of sensors under local and neighbor ESs.*

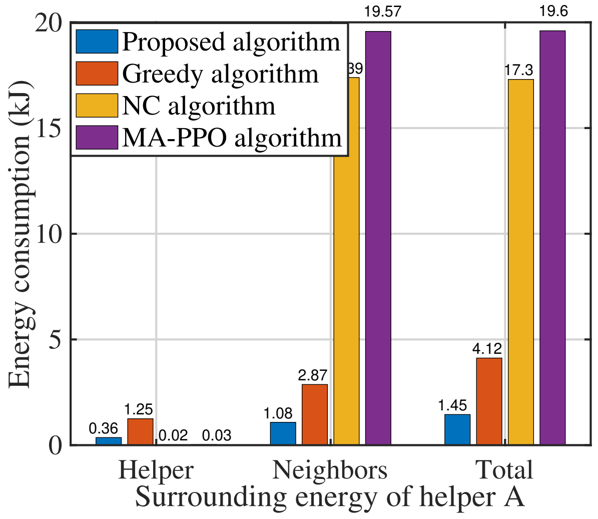


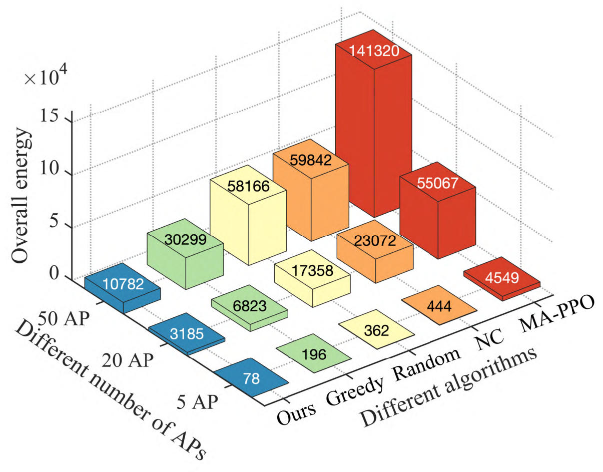
Figure 5. *Contribution of A sacrifices for its surrounding energy.*

- **MA-PPO:** It is a well-known MA-DRL algorithm based on centralized training process. Existing works have used it to develop resource allocation strategies [26].
- **Greedy, Random and Non-cooperative (NC) algorithms:** The Greedy algorithm [27] sets the ESs with more than the average sensors to be helpers ("strong ESs"). The Random algorithm [27] randomly sets ESs to be helpers, while NC sets no ESs to be helpers.

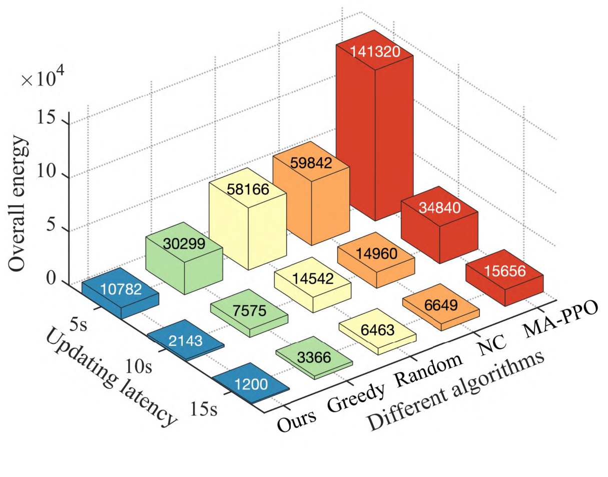
Figure 3 gives the convergency performances of the traditional MA-PPO and the proposed algorithm. It shows that the energy consumption of MA-PPO is approximately six times larger than the proposed algorithm when reaching convergencies, under different ES numbers and updating latency requirements. It is due to two reasons. First, the proposed algorithm greatly reduces the dimension of critic network, by condensing its input into most valuable information. This makes it easy to train the critic network. Second, neighboring information is provided to the actor network, which makes it easier to generate a good action that is beneficial for the whole community.

Figure 2 gives the mean cooperative identities of different ESs under different algorithms. It proves the inefficacy of MA-PPO in Figure 2a, where no ES chooses to be helper, no matter of whether it is proper for it to become a helper of its neighbors or not. On the contrary, both of the Greedy algorithms and proposed algorithm have multiple helpers, as shown in Figure 2b and Figure 2c, respectively. Specifically, Figure 4 shows that ESs with more local sensors while its neighbors having less sensors, more tend to become helpers. This mode can bring down system energy as explained in Subsection 4.5. Moreover, the proposed algorithm has much less helpers than the Greedy, according to Figure 2b and Figure 2c. Remember that a helper would sacrifice its own energy to reduce its surrounding energy (Subsection 3.3). It illustrates that the proposed algorithm learns to sacrifice ES with a higher efficiency.

We take the ES A (pointed out in Figure 2) as an example to show the effectiveness of the helper sacrifice



(a) *Different AP numbers.*



(b) *Different updating latencies.*

Figure 6. *System energies under different number of ESs and updating latencies in different algorithms.*

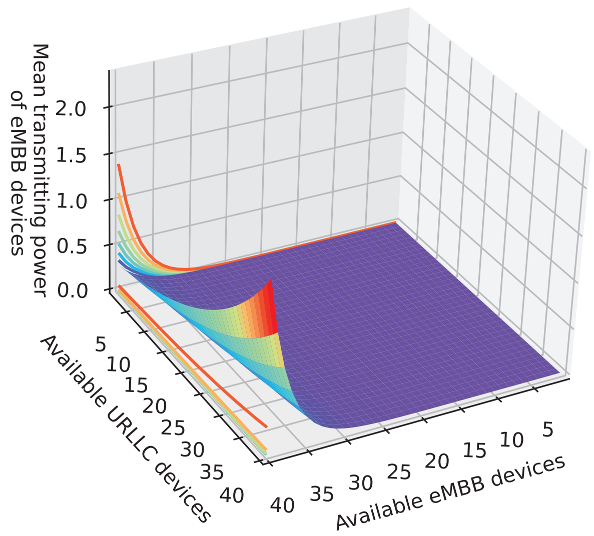


Figure 7. *Mean transmitting power of eMBB sensors under different number of available heterogenous sensors.*

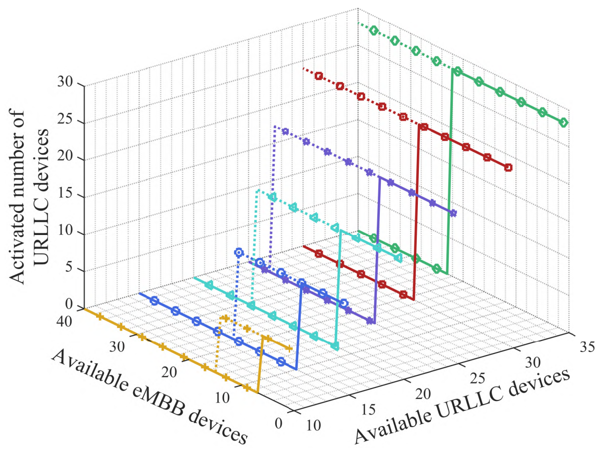


Figure 8. *Mean number of activated URLLC sensors under different number of available heterogenous sensors. Solid line represents $k_i^{\mathrm{U}} = 1, k_i^{\mathrm{B}} = 5$, while dotted line represents $k_i^{\mathrm{U}} = 1, k_i^{\mathrm{B}} = 1$.*

behavior. Figure 5 sequentially gives the mean energy consumption of A, its neighbors (red circle in Figure 2) and the sum of two. When a “strong ES” becomes a helper, it can sacrifice its own energy to reduce the total energy of the surrounding area. In Figure 5, A’s energy consumption (360 J) in the proposed algorithm is much larger than the NC and MA-PPO algorithms (20 J and 30 J). However, it significantly reduces A’s neighbors energy to 1080 J compared with other algorithms. As a result, it leads to a smaller total surrounding energy (1450 J). The Greedy algorithm also sacrifices helper A to reduce its neighboring energy, whose total surrounding energy is smaller than NC and MA-PPO. However, in the Greedy algorithm, all of the “strong ESs”, no matter whether their neighbors have abundant sensors or not, are appointed as helpers. It leads to a low sacrifice efficiency and high neighboring energy.

Figure 6 compares the mean overall system energy in different algorithms, under different numbers of APs/ESs and convergence latency thresholds. Specifically, different numbers of APs represents different resolutions and scales of the DT-WA system map. The proposed algorithm can obtain a smaller system energy than other algorithms under different network scales. Additionally, different DT-WA applications may have different requirements on the convergence latency. The figure shows that the proposed algorithm can achieve different latency requirements with the minimum energy consumption compared with other algorithms. Its well energy performance is due to the effective ESs cooperating process. That is, the proposed algorithm has a higher sacrifice efficiency than the Greedy, and thus has a smaller energy consumption. By comparison, the Random algorithm sometimes generates proper cooperation strategy, which makes its average energy a little bit better than NC algorithm. The MA-PPO keeps to generate poor cooperation strategies, which makes it behave even worse than the NC algorithm.

Figure 7 gives the mean transmitting power of eMBB sensors under different number of available heterogenous sensors. On the one hand, more URLLC sensors would lead to more punctures on eMBB traffics, which would reduce the time-frequency resources for eMBB traffics, and thus increase eMBB transmission delay. In this case, eMBB sensors would emit more transmitting power to keep its transmission latency acceptable. On the other hand, eMBB sensors evenly divide the wireless bandwidth for transmission. A large number of eMBB sensors would have less available bandwidth for each eMBB sensor. Thus, the transmission power needs to increase to keep the transmission rate and reduce transmission latency.

Figure 8 gives the mean number of activated URLLC sensors under different number of available heterogenous sensors. Consistent to the insight mentioned in Subsection 5.3, the mean activated number approaches to zero when the number of available URLLC sensors is small and eMBB is large, and approaches the maximum number (available number) when the opposite situation happens. It is because that, in updating sub-models, the optimal training iteration times for URLLC-based child network and mixed eMBB-URLLC-based integrated learning network becomes zeros when the number of available URLLC sensors is smaller than eMBB sensors to a degree which is determined by the importance factor of k_i^{B} and k_i^{U} in the updating convergency rate. Moreover, when k_i^{U} is larger than k_i^{B}, it means the URLLC data is more important for the sub-model, which would require more URLLC sensors to activate, and it makes the dotted lines more wider on the head than the solid lines.

VII. CONCLUSION

The paper closes the gap of studying the updating process for the AI model in DT-WA. It proposes a distributed cooperation and data collection scheme to solve the energy challenge in the updating process. In the scheme, ESs can share their updated sub-model parameters with neighbors to reduce the latter updating workloads and energy consumptions. Each ES needs to adaptively determine its cooperative identity, activate heterogeneous sensors and allocate wireless and computing resources to guarantee its local and neighbors updating performances. To minimize the overall updating energy, we formulate an optimization problem under updating convergency and latency constraints. Then, a distributed cooperation and resource allocation algorithm is proposed. First, the original problem is transformed into a constraint-MG. Then, the state and action spaces are reduced and several constraint-released methods are proposed. Finally, a distributed large-scale MA-DRL is designed by introducing the MF theory to optimize the ESs actions. Simulation shows the proposed cooperation and data collection scheme can effectively reduce the updating energy for AI model in DT-WA compared with baselines.

ACKNOWLEDGEMENT

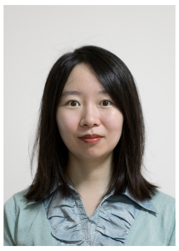
This work was supported by National Key Research and Development Program of China (2020YFB1807900).

References

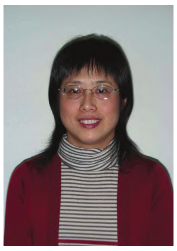
[1] L. U. Khan, W. Saad, *et al.*, "Digital-twin-enabled 6g: Vision, architectural trends, and future directions," *IEEE Communications Magazine*, vol. 60, no. 1, pp. 74–80, 2022.
[2] G. White, A. Zink, *et al.*, "A digital twin smart city for citizen feedback," *Cities*, vol. 110, p. 103064, 2021.
[3] T. Deng, K. Zhang, *et al.*, "A systematic review of a digital twin city: A new pattern of urban governance toward smart cities," *Journal of Management Science and Engineering*, vol. 6, no. 2, pp. 125–134, 2021.
[4] M. Ghaith, A. Yosri, *et al.*, "Synchronization-enhanced deep learning early flood risk predictions: The core of data-driven city digital twins for climate resilience planning," *Water*, vol. 14, no. 22, p. 3619, 2022.
[5] L. Buonocore, J. Yates, *et al.*, "A proposal for a forest digital twin framework and its perspectives," *Forests*, vol. 13, no. 4, p. 498, 2022.

[6] R. Balica, *et al.*, "Machine and deep learning technologies, wireless sensor networks, and virtual simulation algorithms in digital twin cities," *Geopolitics, History, and International Relations*, vol. 14, no. 1, pp. 59–74, 2022.
[7] Y. Xiao, L. Xiao, *et al.*, "Reinforcement learning based energy-efficient collaborative inference for mobile edge computing," *IEEE Transactions on Communications*, vol. 71, no. 2, pp. 864–876, 2022.
[8] W. Y. B. Lim, Z. Xiong, *et al.*, "Hierarchical incentive mechanism design for federated machine learning in mobile networks," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 9575–9588, 2020.
[9] Y. Lu, X. Huang, *et al.*, "Communication-efficient federated learning for digital twin edge networks in industrial iot," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 8, pp. 5709–5718, 2020.
[10] L. Jiang, H. Zheng, *et al.*, "Cooperative federated learning and model update verification in blockchain-empowered digital twin edge networks," *IEEE Internet of Things Journal*, vol. 9, no. 13, pp. 11 154–11 167, 2021.
[11] W. Sun, P. Wang, *et al.*, "Dynamic digital twin and distributed incentives for resource allocation in aerial-assisted internet of vehicles," *IEEE Internet of Things Journal*, vol. 9, no. 8, pp. 5839–5852, 2021.
[12] T. Liu, L. Tang, *et al.*, "Digital-twin-assisted task offloading based on edge collaboration in the digital twin edge network," *IEEE Internet of Things Journal*, vol. 9, no. 2, pp. 1427–1444, 2021.
[13] L. Guangyi, D. Juan, *et al.*, "Native intelligence for 6g mobile network: Technical challenges, architecture and key features," *The Journal of China Universities of Posts and Telecommunications*, vol. 29, no. 1, p. 27, 2022.
[14] M. Jacobellis and M. Ilbeigi, "Digital twin cities: Data availability and systematic data collection," in *Construction Research Congress 2022*, pp. 437–444.
[15] F. Tao, B. Xiao, *et al.*, "Digital twin modeling," *Journal of Manufacturing Systems*, vol. 64, pp. 372–389, 2022.
[16] M. Austin, P. Delgoshaei, *et al.*, "Architecting smart city digital twins: Combined semantic model and machine learning approach," *Journal of Management in Engineering*, vol. 36, no. 4, p. 04020026, 2020.
[17] M. Darabi, V. Jamali, *et al.*, "Hybrid puncturing and superposition scheme for joint scheduling of urllc and embb traffic," *IEEE Communications Letters*, vol. 26, no. 5, pp. 1081–1085, 2022.
[18] Y. Polyanskiy, H. V. Poor, *et al.*, "Channel coding rate in the finite blocklength regime," *IEEE Transactions on Information Theory*, vol. 56, no. 5, pp. 2307–2359, 2010.
[19] A. Brighente, J. Mohammadi, *et al.*, "Interference prediction for low-complexity link adaptation in beyond 5g ultra-reliable low-latency communications," *IEEE Transactions on Wireless Communications*, vol. 21, no. 10, pp. 8403–8415, 2022.
[20] P. Popovski, K. F. Trillingsgaard, *et al.*, "5g wireless network slicing for embb, urllc, and mmtc: A communication-theoretic view," *IEEE Access*, vol. 6, pp. 55 765–55 779, 2018.
[21] Y. Chen, F. Zhao, *et al.*, "Dynamic task offloading for mobile edge computing with hybrid energy supply," *Tsinghua Science and Technology*, vol. 28, no. 3, pp. 421–432, 2022.
[22] J. Wu, R. Li, *et al.*, "Toward native artificial intelligence in 6g networks: System design, architectures, and paradigms," *arXiv Preprint arXiv:2103.02823*, 2021.
[23] J. Schulman, F. Wolski, *et al.*, "Proximal policy optimization algorithms," *arXiv Preprint arXiv:1707.06347*, 2017.
[24] Y. Yang, R. Luo, *et al.*, "Mean field multi-agent reinforcement learning," in *International Conference on Machine Learning*, pp. 5571–5580. PMLR, 2018.
[25] G. TS, "Evolved universal terrestrial radio access (e-utra). further advancements for e-utra physical layer aspects (release 9)," 3GPP TR 36.814, *Tech. Rep.*, 2010.
[26] Y. M. Park, S. S. Hassan, *et al.*, "Joint trajectory and resource optimization of mec-assisted uavs in sub-thz networks: A resources-based multi-agent proximal policy optimization drl with attention mechanism," *arXiv Preprint arXiv:2209.07228*, 2022.
[27] W. Zhan, C. Luo, *et al.*, "Deep-reinforcement-learning-based offloading scheduling for vehicular edge computing," *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 5449–5465, 2020.

Biographies


Mancong Kang received the B.E. degree in electronics information science and technology from the Xi'an University of Posts and Telecommunications, China, in 2017. She is currently working toward the Ph.D. degree with the School of Information and Communication Engineering at Beijing University of Posts and Telecommunications (BUPT) in Beijing, China. Her current research interests include future wireless networks, digital twin, tactile internet, machine learning, and mobile edge computing.

Xi Li received her B.E. degree and Ph.D. degree from Beijing University of Posts and Telecommunications (BUPT) in the major of communication and information system in 2005 and 2010 respectively. In 2017 and 2018, she was a Visiting Scholar with the University of British Columbia, Vancouver, BC, Canada. She is currently a Professor with the School of Information and Communication Engineering of BUPT. She has published more than 100 papers in International journals and conferences. Her current research interests include resource management and intelligent networking in next generation networks, the Internet of Things, and cloud computing. She has also served as a TPC Member of IEEE WCNC 2012/2014/ 2015/2016/2019/2020/2021, PIMRC 2012/2017/2018/2019/2020, GLOBECOM 2015/2017/2018, ICC 2015/2016/2017/2018/2019, Infocom 2018, and CloudCom 2013/2014/2015, the Chair of Special Track on cognitive testbed in CHINACOM 2011, the Workshop Chair of IEEE GreenCom 2019, and a Peer Reviewer of many academic journals.



Hong Ji received the B.S. degree in communications engineering and the M.S. and Ph.D. degrees in information and communications engineering from the Beijing University of Posts and Telecommunications (BUPT), Beijing, China, in 1989, 1992, and 2002, respectively. In 2006, she was a Visiting Scholar with the University of British Columbia, Vancouver, BC, Canada. She is currently a Professor with BUPT. She has authored more than 300 journal/conference papers. Several of her papers had been selected for Best Paper. Her research interests include wireless networks and mobile systems, including cloud computing, machine learning, intelligent networks, green communications, radio access, ICT applications, system architectures, management algorithms, and performance evaluations. She has guest-edited *International Journal of Communication Systems,* (Wiley) Special Issue on Mobile Internet: Content, Security and Terminal. She has served as the Co-Chair for Chinacom'11, and a member of the Technical Program Committee of WCNC, Globecom, ISCIT, CITS, WCSP, ICC, ICCC, PIMRC, IEEE VTC, and Mobi-World. She served on the editorial boards of the *IEEE Transactions on Green Communications and Networking*, *Wiley International Journal of Communication Systems*.

Heli Zhang received the B.S. degree in communication engineering from Central South University in 2009, and the Ph.D. Degree in communication and information System from Beijing University of Posts and Telecommunications (BUPT) in 2014. From 2014 to 2018, she was the Lecturer in the School of Information and Communication Engineering at BUPT. From 2018, she has been the associate professor in the School of Information and Communication Engineering at BUPT. She has been the reviewer for Journals of *IEEE Wireless Communications, IEEE Communication Magazine, IEEE Transactions on Vehicular Technology, IEEE Communication Letters* and *IEEE Transactions on Networking*. She participated in many National projects funded by National Science and Technology Major Project, National 863 High-tech and National Natural Science Foundation of China, and cooperated with many Corporations in research. Her research interests include heterogeneous networks, long-term evolution/fifth generation and Internet of Things.