

Received June 8, 2021, accepted June 16, 2021, date of publication June 25, 2021, date of current version July 8, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3092381

Deep Reinforcement Learning for Task Offloading in Edge Computing Assisted Power IoT

JIANGYI HU¹, (Senior Member, IEEE), YANG LI¹, GAOFENG ZHAO¹, (Member, IEEE),
BO XU^{1,2}, (Student Member, IEEE), YIYANG NI^{1,2}, AND HAITAO ZHAO^{1,2}

¹NARI Group Corporation (State Grid Electric Power Institute), Nanjing 210061, China

²Key Laboratory of Ministry of Education in Broadband Wireless Communication and Sensor Network Technology, Nanjing University of Posts and Telecommunications, Nanjing 210003, China

Corresponding author: Yiyang Ni (niyy@jssnu.edu.cn)


This work was supported by the Science and Technology Project of State Grid Corporation of China (SGCC): The Key Technologies for Research on Power 5G Standardization Module Technology under Grant SGZJXT00JSJS2000455.

ABSTRACT Power Internet of Things (PIoT) is a promising solution to meet the increasing electricity demand of modern cities, but real-time processing and analysis of huge data collected by the devices is challengeable due to limited computing capability of devices and long distance from the cloud center. In this paper, we consider the edge computing assisted PIoT where the computing tasks of the devices can be either processed locally by the devices, or offloaded to edge servers. Aiming to maximize the long-term system utility which is defined as a weighted sum of reduction in latency and energy consumption, we propose a novel task offloading algorithm based on deep reinforcement learning, which jointly optimizes task scheduling, transmit power of the PIoT devices, and computing resource allocation of the edge servers. Specifically, the task execution on each edge server is modeled as a queuing system, in which the current queue state may affect the waiting time for the next tasks. The transmit power and computing resource allocation is first optimized, respectively, and then a deep Q-learning algorithm is adopted to make task scheduling decisions. Numerical results show that the proposed method can improve the system utility.

INDEX TERMS Power Internet of Things, smart grid, edge offloading, resource allocation, deep reinforcement learning.

I. INTRODUCTION

Power Internet of Things (PIoT) is the application of the IoT to the smart grid, which can play a role in every aspect of the smart grid, including power generation, transmission, transformation, distribution, and consumption [1]. By integrating resource of communication infrastructure with that of power infrastructure, as well as making use of advanced information and communication technologies, the PIoT allows information exchange among interconnected devices and further improves the unitization efficiency of the power systems [2]. More Specifically, different kinds of sensing devices (e.g., the global positioning system, cameras, infrared sensors, and radio frequency identification devices) and advanced metering infrastructure (AMI) equipments are deployed in different

The associate editor coordinating the review of this manuscript and approving it for publication was Haipeng Yao .

geographical areas, thus the PIoT can sense the physical world, collect monitoring data for processing, and further make smart decisions of energy management schemes.

However, the collected data can be huge and highly heterogeneous, especially when the number of devices is greatly large, which poses stringent requirements on both communication and computational resources. Besides, some real-time demand side management (DSM) schemes [3] are typically delay-sensitive, which makes data processing tasks even harder. Although the computation capability of the smart devices has been improved in recent years, it is still hard to deal with complex tasks under rigorous delay constraints. Besides, the cloud computing technique also cannot meet the low-latency requirements because of long transmission distance and high construction cost.

Thankfully, edge computing, providing intensive computation and storage resources at the network edge, emerges as

a promising solution to this dilemma. By offloading computation tasks to edge servers instead of the cloud server, communication delay and traffic is reduced and user privacy is also enhanced [4], [5]. Edge computing is well suited for data processing and analysis in a PIIoT scenarios. For instance, an edge computing framework for real-time monitoring grid services was proposed in [6], which further formulated an edge server scheduling problem to reduce the task processing cost. Besides, a well-organized data processing scheme for PIIoT was proposed in [7]. Moreover, authors in [8] investigated the message dissemination of edge devices in the smart grid environment, and proposed an efficient data processing scheme. In [9], based on the edge computing framework, some resource management and load balancing policies were proposed to handle smart grid consumers requests. However, for the PIIoT offloading scenarios, task offloading still causes extra communication overhead including energy consumption and transmission latency, compared to local computing on devices. Thus, there is a tradeoff between performance gain and communication overhead and how to make task offloading decisions is a challenging problem.

There have been a lot of research focusing on edge offloading in recent years. The authors in [10] proposed an offloading method based on Markov decision process (MDP) to control task transfer and execution. The work in [11] reduced both the execution latency and energy consumption for one-device offloading scenario. Ren *et al.* in [12] investigated the latency-minimization problem in a multi-device system with joint communication and computation resource allocation. In addition, some strategies were specifically designed for the PIIoT scenarios aiming at minimize the cost of the edge computing system while satisfying the task completion needs [13]–[15]. Although employing edge servers for task computation is a promising approach to reducing latency and energy consumption, it is still a challenge to make efficient offloading decisions due to the complicated communication topology as well as the dynamic system state. Therefore, in order to improve the long-term performance, it is necessary to design an offloading mechanism that can dynamically adapt to the system change, especially with unknown potential action-state space. Deep reinforcement learning (DRL) is thought as a promising approach to solving the long-term optimization [16]–[21]. The authors in [16] and [17] adopted a DRL algorithm to find the optimal task offloading decisions for a variety of vehicular communication modes. An integrated framework based on DRL was proposed in [18] to improve the performance of edge caching. In [19], the authors proposed a DRL algorithm for an IoT device to choose the edge server and determine the offloading rate according to the current battery level. To reduce the action space of DRL, a distributed learning method combined with game theory was proposed in [20], in which each device was designed to perform task offloading independently. In [21], the authors proposed a scheme in which the devices in a similar channel state could share learned Q-table with each other to decrease iterations. Most recently, a novel DRL algorithm called deep

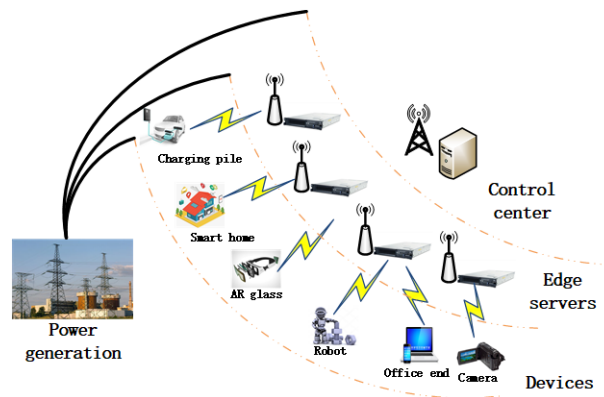


FIGURE 1. The PIIoT system.

deterministic policy gradient is proposed to learn continuous variables [22]–[24]. But none of the mentioned works above give a offloading mechanism which can jointly optimize the task scheduling and transmit power and computing resource allocation to maximize the long-term performance.

In this paper, we consider a PIIoT scenario where the devices compete with each other for sub-channels to access to the edge servers. The task execution on the servers is modeled as a queueing system and the task offloading process of the devices is modeled as an MDP [25]. By jointly optimizing the task scheduling and transmit power and computing resource allocation, our objective is to maximize the system utility, which is a weighted sum of reduction in latency and energy consumption. The main contributions of this study are summarized as follows:

1) We consider a PIIoT system assisted by multiple edge servers and formulate the task offloading problem as a long-term Mixed Integer Non-linear Program (MINLP). We jointly optimize task scheduling and transmit power and computing resource allocation to maximize the system utility defined as a weighted sum of reduction in latency and energy consumption.

2) We decompose the original task offloading problem into (i) a resource allocation problem including the transmit power and computation resource allocation, and (ii) a task scheduling problem. The transmit power and computing resource allocation problem is addressed using optimization methods and then the task scheduling problem is solved using the DRL method.

3) A lot of simulation results are presented and the performance gain of the proposed offloading scheme is validated.

The rest of this paper is organized as follows: The system model and problem formulation are described in section II. A deep Q-learning based task offloading policy is proposed in Section III. Simulation results are presented and discussed in Section IV, followed by the conclusion of this study in Section V.

II. SYSTEM MODEL

In this paper, we consider a PIIoT system assisted by the edge computing, as shown in Fig. 1. The sets of edge access

points (APs) and devices are denoted as $\mathcal{M} = \{1, \dots, M\}$ and $\mathcal{U} = \{1, \dots, U\}$, respectively. Each AP is equipped with an edge server with strong computing capability for the offloaded task processing. For ease of expression, we also denote the set of edge servers as \mathcal{M} . Each device has a computing task which can be computed locally or offloaded to one of the edge servers. In this PloT system, we assume that there has a control center, which can gather the network state from the devices and the servers while making task offloading and resource allocation decisions through a dedicated control channel.

We also assume that the communication topology of this PloT system is invariant within a time frame with fixed length τ [16]. Then the task of device $u \in \mathcal{U}$ generated in each frame is described by a tuple $\pi_u = \{D_u, C_u\}$, where D_u and C_u are the data size [bits] and the workload [cycles] of task u , respectively. The computation to data ratio is given by $\kappa = \frac{C_u}{D_u}$. We divide each frame into three related periods, i.e., task offloading, remote computing, and result return.

- Task offloading. At the beginning of each frame, device u generates a new task π_u . Based on the results of task offloading, the task can be computed locally or sent to one of the edge servers.
- Remote computation. After receiving data from the devices, the edge servers start to execute the their respective tasks. We model each edge server as a queuing system, in which the new received tasks need to wait until the edge server finish the tasks from the former frames.
- Result return. After the remote computation is finished, the edge servers return the computing results to the devices. Similar to the work in [4], due to the size of computing results is much smaller than the size of input tasks, the latency and overhead of returning the computing results is ignored.

A. COMMUNICATION MODEL

The devices access to the edge servers via orthogonal frequency division multiple access scheme such that no intra-cell interference exists in the uplink. Let $\mathcal{N} = \{1, \dots, N\}$ denotes the set of available sub-channels used by each AP. The bandwidth of each sub-channel is $W = B/N$, where B is the total system bandwidth. We use an indicator $a_{u,m}^{t,n}$ to denote the offloading decision of device u , and $a_{u,m}^{t,n} = 1$, if device u offloads to AP m on sub-channel n in frame t ; otherwise, $a_{u,m}^{t,n} = 0$. We assume that each sub-channel can be assigned to at most one device associated with the same AP. Thus, the intra-cell interference is well mitigated, but the inter-cell interference remains among devices. Besides, each device can offload its task to at most one edge server using one sub-channel. Thus, we have the following constraints:

$$\sum_{u=1}^U a_{u,m}^{t,n} \leq 1, \forall n \in \mathcal{N}, \forall m \in \mathcal{M}, \quad (1)$$

and

$$\sum_{n=1}^N \sum_{m=1}^M a_{u,m}^{t,n} \leq 1, \forall u \in \mathcal{U}. \quad (2)$$

The uplink rate of device u to AP m on sub-channel n is given as

$$R_{u,m}^{t,n} = W \log_2 \left(1 + \frac{a_{u,m}^{t,n} p_u^t |h_{u,m}^{t,n}|^2}{\sum_{m' \in \mathcal{M} \setminus \{u\}} \sum_{n' \in \mathcal{N}} a_{u,m'}^{t,n'} |h_{u,m'}^{t,n'}|^2 + \sigma^2} \right), \quad (3)$$

where p_u^t represents the transmit power of device u , σ^2 is the background noise variance, and $h_{u,m}^{t,n}$ denotes the channel gain between device u and AP m on the sub-channel n in frame t .

When device u offload its task to a certain edge server, the related uplink transmission time of device u can be calculated as

$$T_{u,\text{up}}^t = \sum_{m \in \mathcal{M}} \sum_{n \in \mathcal{N}} a_{u,m}^{t,n} \frac{D_u}{R_{u,m}^{t,n}}, \quad (4)$$

and the related energy consumption is given as

$$E_{u,\text{up}}^t = p_u^t T_{u,\text{up}}^t. \quad (5)$$

B. COMPUTATION MODEL

We denote f_u [cycles/s] as the local computing capability of device u . When device u computes its task locally, i.e., $\sum_{m \in \mathcal{M}} \sum_{n \in \mathcal{N}} a_{u,m}^{t,n} = 0$, the local computation time of device u is calculated as

$$T_{u,\text{loc}} = \frac{C_u}{f_u}, \quad (6)$$

and the related energy consumption is given as

$$E_{u,\text{loc}} = P_{u,\text{comp}} T_{u,\text{loc}}, \quad (7)$$

where $P_{u,\text{comp}}$ is the local computing power which is a constant for each device u [20].

If device u offloads its task to server m , the task computing time is

$$T_{u,\text{comp}}^t = \frac{C_u}{F_{m,u}^t}, \quad (8)$$

where $F_{m,u}^t$ [cycles/s] is the server computing capability assigned by server m to device u in frame t .

C. TASK QUEUING MODEL

In the considered PloT system, the task processing on each edge server is modeled as a queuing system in which the queue state in the next frame only depends on the current queue state and task offloading decision. Therefore, we can formulate the queuing model as an MDP. We define the queue state of server m at the beginning of frame t as S_m^t , and we have $S_m^t = \{s_m^1, s_m^2, \dots, s_m^{t-1}\}$, in which s_m^t is the latency for the server m to finish the task received in frame t . We define \mathcal{U}_m^t as the set of devices which offload to server m in frame t .

With the first-come-first-serve policy, the servers need to complete previous tasks first. The latency for server m to finish the tasks received in frame t is defined as

$$s_m^t = s_m^{t-1} + \Delta_m t, \quad (9)$$

where $\Delta_m t$ is the added computing time in frame t , which is defined as

$$\Delta_m t = \arg \max_{u \in \mathcal{U}_m^t} \left(\frac{C_u}{F_{m,u}^t} + \max\{T_{u,\text{up}}^t - s_m^{t-1}, 0\} \right). \quad (10)$$

Similar to the work in [17], we also denote the maximal waiting time of device u in frame t as

$$T_{u,\text{wait}}^t = s_m^{t-1}. \quad (11)$$

Then, the total offloading latency is defined as

$$T_{u,\text{off}}^t = T_{u,\text{comp}}^t + T_{u,\text{up}}^t + T_{u,\text{wait}}^t. \quad (12)$$

D. UTILITY FUNCTION

Finally, the latency and energy consumption can be denoted as

$$\begin{aligned} T_u^t &= \mathbb{I}(\sum_{m \in \mathcal{M}} \sum_{n \in \mathcal{N}} a_{u,m}^{t,n} = 1) T_{u,\text{off}}^t \\ &+ \mathbb{I}(\sum_{m \in \mathcal{M}} \sum_{n \in \mathcal{N}} a_{u,m}^{t,n} = 0) T_{u,\text{loc}}^t, \end{aligned} \quad (13)$$

and

$$\begin{aligned} E_u^t &= \mathbb{I}(\sum_{m \in \mathcal{M}} \sum_{n \in \mathcal{N}} a_{u,m}^{t,n} = 1) E_{u,\text{up}}^t \\ &+ \mathbb{I}(\sum_{m \in \mathcal{M}} \sum_{n \in \mathcal{N}} a_{u,m}^{t,n} = 0) E_{u,\text{loc}}^t, \end{aligned} \quad (14)$$

respectively, where $\mathbb{I}(x)$ is an indicator function which is equal to 1 if x is true; otherwise 0. The utility of device u in frame t is defined as a weighted sum of reduction in latency and energy consumption, which takes the form [4]

$$J_u^t = \beta_1 \frac{T_{u,\text{loc}}^t - T_u^t}{T_{u,\text{loc}}^t} + \beta_2 \frac{E_{u,\text{loc}}^t - E_u^t}{E_{u,\text{loc}}^t}, \quad (15)$$

where $\beta_1, \beta_2 \in [0, 1]$ are two positive weights.

We further define $\mathcal{A}^t = \{a_{u,m}^{t,n} | u \in \mathcal{U}, n \in \mathcal{N}, m \in \mathcal{M}\}$, $\mathcal{P}^t = \{p_u^t | 0 \leq p_u^t \leq P^{\max}, u \in \mathcal{U}_{\text{up}}^t\}$, and $\mathcal{F}^t = \{F_{m,u}^t | 0 \leq F_{m,u}^t \leq F_m, u \in \mathcal{U}_m^t, m \in \mathcal{M}\}$ as the decision sets of task scheduling, transmit power allocation, and computing resource allocation, respectively. In addition, $\mathcal{U}_{\text{up}}^t$ denotes the set of the devices which choose to offload their tasks in frame t . Note that $p_u^t = 0$ and $F_{m,u}^t = 0$, if $u \notin \mathcal{U}_{\text{up}}^t$.

E. PROBLEM FORMULATION AND DECOMPOSITION

We formulate a problem which jointly considers task offloading decision and the resource allocation to maximize the average utility:

$$\max_{\{\Omega^t | 1 \leq t \leq T\}} \frac{1}{T} \sum_{t=1}^T \sum_{u=1}^U J_u^t \quad (16a)$$

$$\text{s.t. } a_{u,m}^{t,n} \in \{0, 1\}, \forall u \in \mathcal{U}, m \in \mathcal{M}, n \in \mathcal{N}, \quad (16b)$$

$$\sum_{u \in \mathcal{U}} \sum_{m \in \mathcal{M}} a_{u,m}^{t,n} \leq 1, \forall n \in \mathcal{N}, \quad (16c)$$

$$\sum_{u \in \mathcal{U}} a_{u,m}^{t,n} \leq 1, \forall m \in \mathcal{M}, n \in \mathcal{N}, \quad (16d)$$

$$0 < p_u^t \leq P^{\max}, \forall u \in \mathcal{U}_{\text{up}}^t, \quad (16e)$$

$$\sum_{u \in \mathcal{U}_m^t} F_{m,u}^t \leq F_m, m \in \mathcal{M}, \quad (16f)$$

$$F_{m,u}^t > 0, \forall u \in \mathcal{U}_m^t, m \in \mathcal{M}. \quad (16g)$$

$$T_{u,\text{up}}^t \leq \tau, \forall u \in \mathcal{U}_{\text{up}}^t. \quad (16h)$$

where $\Omega^t = \{\mathcal{A}^t, \mathcal{P}^t, \mathcal{F}^t\}$ is a group of the three variables. The constraint (16h) suggests that if device u chooses to offload, the transmit time of device u should be smaller than the frame length τ .

It is hard to find the optimal solution to the problem (16) because the problem (16) is an MINLP. Besides, the current decisions of task scheduling and resource allocation will affect the decisions in the next frame, which make the problem (16) more difficult. To deal with this dilemma, we decompose the original problem into two problems, i.e., the task scheduling and resource allocation problems. Specifically, given the decisions of transmit power and computing resource allocation, the task scheduling problem is formulated as $(\mathcal{P}^t, \mathcal{F}^t)$

$$\max_{\mathcal{A}^t} \sum_{u=1}^U J_u^t(\mathcal{P}^{*,t}, \mathcal{F}^{*,t}) \quad (17a)$$

$$\text{s.t. } (16b) - (16g). \quad (17b)$$

Then, given the task scheduling decision \mathcal{A}^t , the transmit power and computing resource allocation problem is formulated as

$$\max_{\mathcal{P}^t, \mathcal{F}^t} \sum_{u=1}^U J_u^t(\mathcal{A}^t) \quad (18a)$$

$$\text{s.t. } (16e) - (16h). \quad (18b)$$

Note that \mathcal{P}^t and \mathcal{F}^t can be individually optimized due to their separation in both the objective function and the constraints. Thus, we further decompose (18) into two sub-problems, i.e.,

$$\min_{\mathcal{P}^t} \sum_{u \in \mathcal{U}_{\text{up}}^t} \frac{\beta_1 T_{u,\text{up}}^t}{T_{u,\text{loc}}^t} + \frac{\beta_2 E_{u,\text{up}}^t}{E_{u,\text{loc}}^t} \quad (19a)$$

$$\text{s.t. } (16e) \text{ and } (16h), \quad (19b)$$

and

$$\min_{\mathcal{F}^t} \sum_{m \in \mathcal{M}} \sum_{u \in \mathcal{U}_m^t} \frac{C_u}{F_{m,u}^t} \quad (20a)$$

$$\text{s.t. } (16f) \text{ and } (16g). \quad (20b)$$

In the next section, we will give the solutions to (17), (19), and (20), respectively, and the solutions finally are formed into the solution to the problem (16).

III. PROPOSED TASK OFFLOADING ALGORITHM

In this section, we first solve the transmit power allocation problem as well as the computing resource allocation problem, given the task scheduling decision. Then we adopt a deep Q-learning algorithm to determine the task scheduling policy.

A. TRANSMIT POWER ALLOCATION

Note that problem (19) is non-convex, since the uplink transmission of one device is effected by the other devices. We first give an approximation of inter-cell interference in which the transmit power in the interference is maximal, and the approximate uplink rate $\hat{R}_{u,m}^{t,n}$ in the objective function of (19) is given by

$$\hat{R}_{u,m}^{t,n} = W \log_2 \left(1 + \frac{a_{u,m}^{t,n} p_u^t |h_{u,m}^{t,n}|^2}{\sum_{m' \in \mathcal{M} \setminus m} \sum_{u' \in \mathcal{U} \setminus u} a_{u',m'}^{t,n} P^{\max} |h_{u',m'}^{t,n}|^2 + \sigma^2} \right). \quad (21)$$

Based on $\hat{R}_{u,m}^{t,n}$, we denote the approximate uplink transmission time and the approximate transmission energy consumption as $\hat{T}_{u,\text{up}}^t$ and $\hat{E}_{u,\text{up}}^t$, respectively. By substituting $\hat{R}_{u,m}^{t,n}$ into problem (19), we can decouple the uplink rate of the devices from each other and reformulate the problem (19) as

$$\min_{p_u^t} \Omega_u^t(p_u^t) \quad (22a)$$

$$\text{s.t. } 0 < p_u^t \leq P^{\max}, \forall u \in \mathcal{U}_{\text{up}}^t \quad (22b)$$

$$\hat{T}_{u,\text{up}}^t \leq \tau, \forall u \in \mathcal{U}_{\text{up}}^t, \quad (22c)$$

$$\text{where } \Omega_u^t(p_u^t) = \frac{\beta_1 \hat{T}_{u,\text{up}}^t}{T_{u,\text{loc}}^t} + \frac{\beta_2 \hat{E}_{u,\text{up}}^t}{E_{u,\text{loc}}^t}.$$

Lemma 1: $\Omega_u^t(p_u^t)$ is quasi-convex in the domain $(0, P^{\max}]$.

proof: $\Omega_u^t(p_u^t)$ and $\Omega_u^{\prime\prime}(p_u^t)$ are the first and second derivative of $\Omega_u^t(p_u^t)$. We have

$$\Omega_u^t(p_u^t) = \frac{\beta_2 \log_2(1 + A_u^t p_u^t) - \frac{A_u^t(\beta_1 + \beta_2 p_u^t)}{(1 + A_u^t p_u^t) \ln 2}}{\log_2^2(1 + A_u^t p_u^t)}, \quad (23)$$

and

$$\Omega_u^{\prime\prime}(p_u) = \frac{A_u^t(C1 + C2)}{\ln 2(1 + A_u^t p_u^t)^2 \log_2^3(1 + A_u^t p_u^t)}, \quad (24)$$

where $C1 = (A_u^t \beta_1 - \beta_2(2 + A_u^t p_u^t)) \log_2(1 + A_u^t p_u^t)$, and $C2 = \frac{2A_u^t(\beta_1 + \beta_2 p_u^t)}{\ln 2}$.

Note that $\Omega_u^t(p_u^t) = 0$, when

$$\Gamma(p_u^{t*}) = \beta_2 \log_2(1 + A_u^t p_u^{t*}) - \frac{A_u^t(\beta_1 + \beta_2 p_u^{t*})}{(1 + A_u^t p_u^{t*}) \ln 2} = 0, \quad (25)$$

where $A_u^t = \frac{h_{u,m}^{t,n}}{(\sum_{m' \in \mathcal{M} \setminus m} \sum_{u' \in \mathcal{U} \setminus u} a_{u',m'}^{t,n} P^{\max} |h_{u',m'}^{t,n}|^2 + \sigma^2)}$, $\beta_1 = \frac{\lambda_1 D_u}{T_{u,\text{loc}} W}$, and $\beta_2 = \frac{\lambda_2 D_u}{E_{u,\text{loc}} W}$. Substituting p_u^* into (24), we have

$$C1 + C2 = A_u^t(\beta_1 + \beta_2 p_u^{t*}) \log_2(1 + A_u^t p_u^{t*}) \geq 0. \quad (26)$$

So we can verify that $\Omega_u^t(p_u^t)$ is quasi-convex in the domain $(0, P^{\max}]$.

Based on Lemma 1, we can confirm that $\Gamma'(p_u) > 0$, and $\Gamma(0) < 0$, which means $\Gamma(p_u)$ is a monotonically increasing function in the domain $(0, P^{\max}]$. Furthermore, the optimal solution p_u^* satisfies $p_u^* = P^{\max}$ or $\Gamma(p_u^*) = 0$. We propose a bisection method with a convergence threshold ξ to solve this problem, and we repeat narrowing the scope between P^{\min} and P^{\max} according to the value of $\Gamma(p_u^t)$. Details are shown in Algorithm 1.

Algorithm 1 Bisection method based transmit power allocation.

- 1: **initialization:** $P^{\min} = 0$.
- 2: **repeat**
- 3: $p_u^t = (P^{\max} + P^{\min})/2$.
- 4: **if** $\Gamma(p_u^t) \leq 0$ **then**
- 5: $p_u^t = P^{\min}$.
- 6: **else**
- 7: $p_u^t = P^{\max}$.
- 8: **end if**
- 9: **until** $P^{\max} - P^{\min} \leq \xi$.
- 10: **output** $p_u^t = (P^{\max} + P^{\min})/2$.

B. COMPUTING RESOURCE ALLOCATION

The objective function of problem (20) is convex, because the elements in its Hessian matrix is $\partial^2 J_u^t / \partial (F_{m,u}^t)^2 = \frac{2C_u}{(F_{m,u}^t)^3} > 0$ or $\partial^2 J_u / \partial F_{m,u}^t \partial F_{m',u'}^t = 0$ ($m \neq m'$). Hence, the Lagrangian function of (20) is

$$L(\mathcal{F}^t, \lambda, \nu) = \sum_{m \in \mathcal{M}} \sum_{u \in \mathcal{U}} \frac{C_u}{F_{m,u}^t} + \sum_{m \in \mathcal{M}} \lambda_m \sum_{u \in \mathcal{U}} (F_{m,u}^t - F_m) + \sum_{m \in \mathcal{M}} \sum_{u \in \mathcal{U}} \nu_{m,u} F_{m,u}^t, \quad (27)$$

where $\lambda = [\lambda_1, \dots, \lambda_M]$ and $\nu = [\nu_{1,1}, \dots, \nu_{M,U}]$ are the vector of Lagrangian multipliers. By evaluating the derivatives for $F_{m,u}^t$ in (27), we can get

$$\frac{\partial L(\mathcal{F}^t, \lambda, \nu)}{\partial F_{m,u}^t} = -\frac{C_u}{(F_{m,u}^t)^2} + \lambda_m + \nu_{m,u}, \forall m \in \mathcal{M}, \forall u \in \mathcal{U}_m^t. \quad (28)$$

Note that $F_{m,u}^{t*}$ satisfies $-\frac{C_u}{(F_{m,u}^{t*})^2} + \lambda_m + \nu_{m,u} = 0$, and we have

$$F_{m,u}^{t*} = \sqrt{D_u / (\lambda_m + \nu_{m,u})}, \forall m \in \mathcal{M}, \forall u \in \mathcal{U}_m^t. \quad (29)$$

Combined with the constraint in (16f), we can obtain the following relationships:

$$\nu_m^* + \nu_{m,u}^* = \left(\frac{1}{F_m} \sum_{u \in \mathcal{U}_m^t} \sqrt{C_u} \right)^2, \forall m \in \mathcal{M}, \forall u \in \mathcal{U}_m^t, \quad (30)$$

and

$$F_{m,u}^{t*} = \frac{F_m \sqrt{C_u}}{\sum_{u \in \mathcal{U}_m^t} \sqrt{C_u}}, \forall m \in \mathcal{M}, \forall u \in \mathcal{U}_m^t. \quad (31)$$

C. DEEP Q-LEARNING BASED TASK SCHEDULING

To obtain the task scheduling decision \mathcal{A}^t in (16), the Q-learning technique is adopted. The queue states of the edge servers at the beginning of frame t is defined as $S^t = \{S_1^t, S_2^t, \dots, S_M^t\}$, which can be evaluated according to (9)-(11). Given the state S^t , the control center determines the task scheduling decision \mathcal{A}^t as well as the resource allocation decisions \mathcal{P}^t and \mathcal{F}^t . Define the instantaneous reward as $J^t = \sum_{u=1}^U J_u^t$ and the offloading experience as $(S^t, \mathcal{A}^t, J^t, S^{t+1})$. We use a Q-function $Q(S^t, \mathcal{A}^t)$ to denote the Q-reward, and the updated rule is written as

$$Q(S^t, \mathcal{A}^t) \leftarrow (1 - \varphi)Q(S^t, \mathcal{A}^t) + \varphi \left(J^t + \gamma \max_{\mathcal{A}^{t+1}} Q(S^{t+1}, \mathcal{A}^{t+1}) \right), \quad (32)$$

where $\varphi \in [0, 1]$ is the weight of the current offloading utility, and $\gamma \in [0, 1]$ is the myopic view about the future reward of the devices. We apply the ϵ -greedy policy with exploration probability $\epsilon \in [0, 1]$ to avoid Q-learning staying in the local maximum, and keep the balance between exploration and exploitation. The long-term evaluated \mathcal{A}^t and S^t are stored in the Q-table.

However, the size of the Q-table can be huge especially when the considered problems are complex and the states are massive. To overcome this challenge, we propose a deep Q-learning based task offloading algorithm which use a neural network, instead of the Q-table, to capture the complex relationship between the server state and the scheduling decision. The main deep Q-network is denoted as $Q(S^t, \mathcal{A}^t; \theta)$, where θ is the parameter vector for the main neural network. The input of the neural network is the server state, and the output is the Q-reward. We adopt a target network with the parameter vector θ_{tar} to generate the loss of each action during the training. The structure of the target network is as same as the main deep Q-network.

Let Y_{tar}^t be the output of the target network, we can define the loss function of deep Q-learning in each iteration as

$$L(\theta) = E[Y_{\text{tar}}^t - Q(S^t, \mathcal{A}^t; \theta)]. \quad (33)$$

In order to mitigate the over-estimation issue of the Q-value in (33), we adopt the double deep Q-learning [26], in which the Y_{tar}^t is given by

$$Y_{\text{tar}}^t = J^t + Q\left(S^{t+1}, \arg \max_{\mathcal{A}^t} Q(S^t, \mathcal{A}^t; \theta); \theta_{\text{tar}}\right). \quad (34)$$

We use an experience replay to store the former offloading experience in deep Q-learning, and then randomly select sample batches from the replay for training, so as to overcoming the shortcomings caused by the sample correlation. We update the parameter vector θ_{tar} every K times with a scalar ζ , and we have

$$\theta_{\text{tar}} = \zeta \theta + (1 - \zeta) \theta_{\text{tar}}. \quad (35)$$

Combined with the resource allocation, the proposed deep Q-learning approach is described in Algorithm 2.

Algorithm 2 Deep Q-learning for task offloading.

- 1: **Initialization:** Initialize the experience replay buffer, parameter vectors θ and θ_{tar} .
 - 2: **for** episode $i = 1, \dots, I$ **do**
 - 3: Receive the initial state S^1 with empty task queue.
 - 4: **for** Frame $t = 1, \dots, T$ **do**
 - 5: Generate a random $\rho \in [0, 1]$
 - 6: **if** $\rho \leq \epsilon$ **then**
 - 7: Choose \mathcal{A}^t randomly;
 - 8: **else**
 - 9: $\mathcal{A}^t = \arg \max Q(S^t, \mathcal{A}^t; \theta)$.
 - 10: **end if**
 - 11: Given \mathcal{A}^t , evaluate the results of resource allocation according to **Algorithm 1** and (31).
 - 12: Execute action \mathcal{A}^t , and derive the current reward J^t and the next state S^{t+1} .
 - 13: Store the experience $(S^t, \mathcal{A}^t, J^t, S^{t+1})$ into the experience replay buffer.
 - 14: Choose a batch of samples from the experience replay buffer randomly, and perform a gradient decent algorithm on (33).
 - 15: Every K steps, update the target network by (35).
 - 16: **end for**
 - 17: **end for**
-

IV. NUMERICAL RESULTS

In this section, we conduct experiments to validate the performance of the proposed algorithm in terms of average utility with respect to the MU number, frame length, task workload and so on. All codes are implemented in python 3.7 with TensorFlow 2.0 and are run on a computer with CPU i7-7700.

We consider a multi-server scenario where $M = 7$ and the distance between two neighboring APs is set as 1 km. The path loss is modeled as $L[\text{dB}] = 140.7 + 36.7 \log_{10} d_{[\text{km}]}$ [4]. The total bandwidth is $B = 20$ MHz and the sub-channel number is equal to the number of the devices per cell. The task size of each device is 420Kb [27] and the maximal transmit power is 0.1W. We consider a pdf2text application with $\kappa = 8900$ cycles/byte [20]. We use a fully-connected deep neural network composed of one layer with 100 hidden neurons to build the main network and the target network, respectively. More default parameter values are listed in Table 1.

For performance comparison, we introduce three baseline algorithms, i.e., joint offloading and resource allocation (JORA) algorithm [4], greedy offloading and joint resource allocation (GORA) algorithm [28], and random offloading and joint resource allocation (RORA) algorithm. In the JORA algorithm, the control center makes joint task offloading and resource allocation in each frame. In the GORA algorithm, the devices offload the tasks to the edge servers for processing as possible and greedily access to the sub-channels according to the channel gain until the system utility does not increase. In the RORA algorithm, the devices randomly choose local computing or task offloading.

TABLE 1. Default simulation parameters.

Parameter	Value
AP number M	7
Number of devices U	28
Frame length τ	2 s
Server computing capability $F_m, \forall m \in \mathcal{M}$	10^{10} cycles/s
Local computing capability $f_u, \forall u \in \mathcal{U}$	10^9 cycles/s
Data size $D_u, \forall u \in \mathcal{U}$	420 KB
Local computing power $P_{u,comp}, \forall u \in \mathcal{U}$	0.8 W
Maximal transmit power P^{\max}	0.1 W
Background noise variance σ^2	-100 dBm
Computation to data ratio κ	8900 Cycles/bit
Weights β_1 and β_2	0.8 and 0.2
Replace cycles K	320
Batch size	64
Learning rate	0.1
Exploration probability	0.9

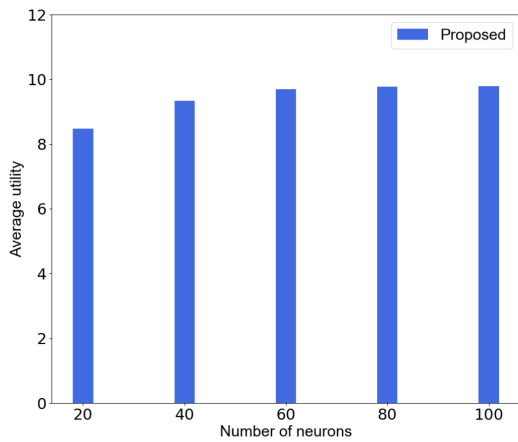


FIGURE 2. Comparison of average utility against different number of neurons.

However, the three baseline algorithms do not take the server state transition into account.

We first show the effect of the neuron number on the performance of the proposed algorithm in Fig. 2, where neurons number varies from 20 to 100. It is shown that more neurons lead to better performance. However, the increase tends to be slow when the neuron number is large, which suggests that the proposed algorithm can achieve satisfying performance with a reasonable number of neurons, not always a huge number.

In Fig. 3, we compare the system average utility of the proposed algorithm with the baseline algorithms with respect to the number of devices. On the whole, we can find that the proposed deep Q-learning based algorithm can achieve better performance than the baseline algorithms, which ignore the effects of the current action on the following actions. Moreover, it is found that the average utility increases with the number of devices when the number of devices is small. This is because the utility is defined as a weighted sum of reduction in latency and energy consumption, and when

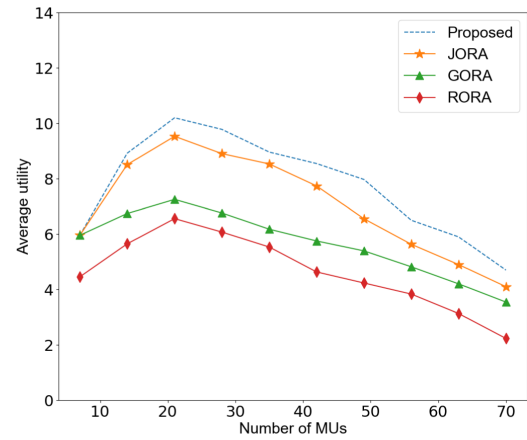


FIGURE 3. Comparison of average utility against different numbers of the devices with $D_u = 420$ Kb, $\kappa = 8900$, $\tau = 2$ s, and $F_m = 10$ GHz.

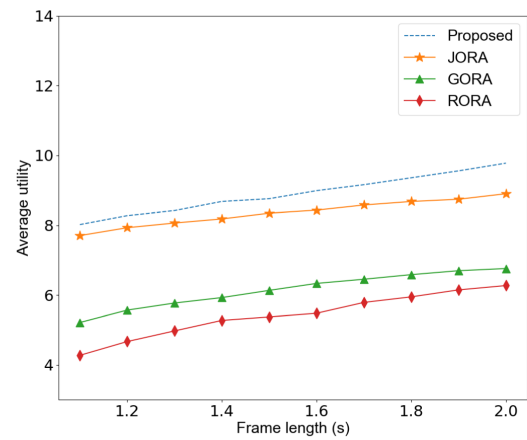


FIGURE 4. Comparison of average utility against different frame length with $U = 28$, $D_u = 420$ Kb, $\kappa = 8900$, and $F_m = 10$ GHz.

the number of devices is not large, the edge servers have enough computing capability to serve more devices which acquire more benefits from offloading compared with local computing. However, due to limited bandwidth resources, when the number of the devices is relatively large, the uplink transmission time of each device is significantly increased resulting in the decreased average utility.

In Fig. 4, we compare the proposed algorithm with the baseline algorithms over different values of frame length τ . We find that the proposed algorithm still achieves better performance than the baseline algorithms. Furthermore, it is found that the average utility increases with τ . This is because given a larger τ , the servers have more time to process tasks in its queue, which can significantly reduce the waiting time and the latency of offloading.

To further investigate the performance of the proposed algorithm, we present the effects of the bandwidth, the data size, the task workload, and the server computing capability in Figs. 5(a), 5(b), 5(c), and 5(d), respectively. Results show that the proposed algorithm can achieve From Fig. 5(a),

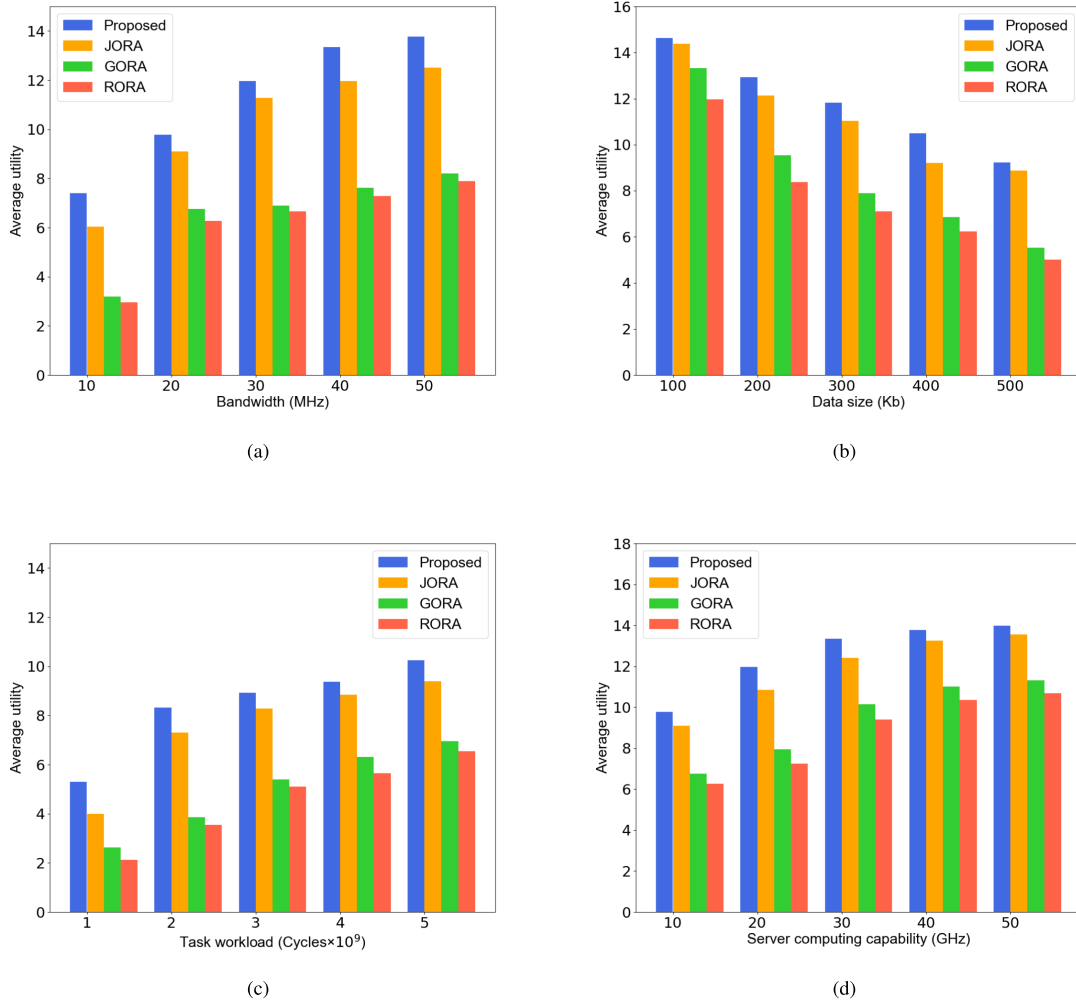


FIGURE 5. Comparison of average utility against different bandwidth, data size, task workloads, and server computation capabilities with $U = 28$ and $\tau = 2$ s.

we can see that the average utility increases with B , this is because the bandwidth of each sub-channel is equally allocated and a larger bandwidth can reduce the transmit power and uplink latency. From Fig. 5(b), since the data size directly affects the uplink latency, we can see that the average utility decreases with D_u . However, in Fig. 5(c), we note that the average utility increases with the task size. The reason for this fact is that the utility function is determined by the reduction of the latency and the energy consumption, thus offloading huge tasks to the edge servers can save more latency and energy compared with local computing. Finally, According to Fig. 5(d), it is seen that the average utility increases as F_m increases. This implies that when the computing capability is strong, more tasks can be computed in the current frame. In summary, the average utility of the proposed algorithm is always better than those of the baseline algorithms.

V. CONCLUSION

In this paper, we considered a PloT system with multiple edge servers and modeled the task processing on each edge

server as a queuing system. A task offloading problem was formulated, which aimed to maximize the system average utility defined as a weighted sum of reduction in latency and energy consumption. We decomposed the original problem into a task scheduling problem and a resource allocation problem. A bisection method was used to find the solution to the transmit power allocation problem and a closed-form solution to the computing resource allocation is obtained. Moreover, we adopted the deep Q-learning method to learn the task scheduling policy. The simulation results demonstrated that the proposed algorithm can improve the system performance. To implement the proposed algorithm in a PloT system, we only need to deploy and run the proposed algorithm in a control center. Specifically, in each frame the edge servers collect the necessary information and upload it to the control center. With the collected information, the control center runs the proposed algorithm and broadcasts the decisions to the edge servers to perform the appropriate actions. The convenience of deployment and implementation makes the proposed algorithm more practical.

REFERENCES

- [1] Q. Ou, Y. Zhen, X. Li, Y. Zhang, and L. Zeng, "Application of Internet of Things in smart grid power transmission," in *Proc. 3rd FTRA Int. Conf. Mobile, Ubiquitous, Intell. Comput.*, Vancouver, BC, Canada, Jun. 2012, pp. 96–100.
- [2] Q. Wang and Y. G. Wang, "Research on power Internet of Things architecture for smart grid demand," in *Proc. 2nd IEEE Conf. Energy Internet Energy Syst. Integr. (EI2)*, Oct. 2018, pp. 1–9.
- [3] Y. Liu, R. Yu, M. Pan, Y. Zhang, and S. Xie, "SD-MAC: Spectrum database-driven MAC protocol for cognitive machine-to-machine networks," *IEEE Trans. Veh. Technol.*, vol. 66, no. 2, pp. 1456–1467, Feb. 2017.
- [4] T. X. Tran and D. Pompili, "Joint task offloading and resource allocation for multi-server mobile-edge computing networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 1, pp. 856–868, Jan. 2019.
- [5] S. A. A. Shah, E. Ahmed, M. Imran, and S. Zeadally, "5G for vehicular communications," *IEEE Commun. Mag.*, vol. 56, no. 1, pp. 111–117, Jan. 2018.
- [6] Y. Huang, Y. Lu, F. Wang, X. Fan, J. Liu, and V. C. M. Leung, "An edge computing framework for real-time monitoring in smart grid," in *Proc. IEEE Int. Conf. Ind. Internet (ICII)*, Bellevue, WA, USA, Oct. 2018, pp. 99–108.
- [7] T. Islam and M. M. A. Hashem, "A big data management system for providing real time services using fog infrastructure," in *Proc. IEEE Symp. Comput. Appl. Ind. Electron. (ISCAIE)*, Penang Island, Malaysia, Apr. 2018, pp. 85–89.
- [8] N. Kumar, S. Zeadally, and J. J. P. C. Rodrigues, "Vehicular delay-tolerant networks for smart grid data management using mobile edge computing," *IEEE Commun. Mag.*, vol. 54, no. 10, pp. 60–66, Oct. 2016.
- [9] S. Zahoor, N. Javaid, A. Khan, B. Ruqia, F. J. Muhammad, and M. Zahid, "A cloud-fog-based smart grid model for efficient resource utilization," in *Proc. 14th Int. Wireless Commun. Mobile Comput. Conf. (IWCMC)*, Limassol, Cyprus, Jun. 2018, pp. 1154–1160.
- [10] J. Liu, Y. Mao, J. Zhang, and K. B. Letaief, "Delay-optimal computation task scheduling for mobile-edge computing systems," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Barcelona, Spain, Jul. 2016, pp. 1451–1455.
- [11] Y. Mao, J. Zhang, and K. B. Letaief, "Dynamic computation offloading for mobile-edge computing with energy harvesting devices," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 12, pp. 3590–3605, Dec. 2016.
- [12] J. Ren, G. Yu, Y. Cai, and Y. He, "Latency optimization for resource allocation in mobile-edge computation offloading," *IEEE Trans. Wireless Commun.*, vol. 17, no. 8, pp. 5506–5519, Aug. 2018.
- [13] C. Yang, X. Chen, Y. Liu, W. Zhong, and S. Xie, "Efficient task offloading and resource allocation for edge computing-based smart grid networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Shanghai, China, May 2019, pp. 1–6.
- [14] J. Yao, Z. Li, Y. Li, J. Bai, J. Wang, and P. Lin, "Cost-efficient tasks scheduling for smart grid communication network with edge computing system," in *Proc. 15th Int. Wireless Commun. Mobile Comput. Conf. (IWCMC)*, Tangier, Morocco, Jun. 2019, pp. 272–277.
- [15] M. Li, L. Rui, X. Qiu, S. Guo, and X. Yu, "Design of a service caching and task offloading mechanism in smart grid edge network," in *Proc. IEEE Int. Wireless Commun. Mobile Comput. Conf. (IWCMC)*, Tangier, Morocco, Jun. 2019, pp. 249–254.
- [16] K. Zhang, S. Leng, X. Peng, L. Pan, S. Maharjan, and Y. Zhang, "Artificial intelligence inspired transmission scheduling in cognitive vehicular communications and networks," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1987–1997, Apr. 2019.
- [17] K. Zhang, Y. Zhu, S. Leng, Y. He, S. Maharjan, and Y. Zhang, "Deep learning empowered task offloading for mobile edge computing in urban informatics," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 7635–7647, Oct. 2019.
- [18] Y. He, N. Zhao, and H. Yin, "Integrated networking, caching, and computing for connected vehicles: A deep reinforcement learning approach," *IEEE Trans. Veh. Technol.*, vol. 67, no. 1, pp. 44–55, Jan. 2018.
- [19] M. Min, L. Xiao, Y. Chen, P. Cheng, D. Wu, and W. Zhuang, "Learning-based computation offloading for IoT devices with energy harvesting," *IEEE Trans. Veh. Technol.*, vol. 68, no. 2, pp. 1930–1941, Feb. 2019.
- [20] T. Q. Dinh, Q. D. La, T. Q. S. Quek, and H. Shin, "Learning for computation offloading in mobile edge computing," *IEEE Trans. Wireless Commun.*, vol. 66, no. 12, pp. 6353–6367, Dec. 2018.
- [21] Y. Luo, M. Zeng, and H. Jiang, "Learning to tradeoff between energy efficiency and delay in energy harvesting-powered D2D communication: A distributed experience-sharing algorithm," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 5585–5594, Jun. 2019.
- [22] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," in *Proc. ICML*, Beijing, China, 2014, pp. 1–9.
- [23] Y. Dai, D. Xu, S. Maharjan, Z. Chen, Q. He, and Y. Zhang, "Blockchain and deep reinforcement learning empowered intelligent 5G beyond," *IEEE Netw.*, vol. 33, no. 3, pp. 10–17, May 2019.
- [24] C. Qiu, Y. Hu, Y. Chen, and B. Zeng, "Deep deterministic policy gradient (DDPG)-based energy harvesting wireless communications," *IEEE Internet Things J.*, vol. 6, no. 5, pp. 8577–8588, Oct. 2019.
- [25] M. Abu Alsheikh, D. T. Hoang, D. Niyato, H.-P. Tan, and S. Lin, "Markov decision processes with applications in wireless sensor networks: A survey," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 3, pp. 1239–1267, 3rd Quart., 2015.
- [26] H. Van Hasselt, A. Guez, and D. Silver, "Deep reinforcement learning with double Q-learning," in *Proc. CORR*, Phoenix, AZ, USA, Feb. 2016, pp. 2094–2100.
- [27] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Trans. Netw.*, vol. 24, no. 5, pp. 2795–2808, Oct. 2016.
- [28] S. Sardellitti, G. Scutari, and S. Barbarossa, "Joint optimization of radio and computational resources for multicell mobile-edge computing," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 1, no. 2, pp. 89–103, Jun. 2015.



JIANGYI HU (Senior Member, IEEE) received the Ph.D. degree in power system and automation from Shanghai Jiaotong University. He is a researcher level Senior Engineer, currently serving as the Director, the General Manager, and the Deputy Secretary of Party Committee of NARI Group Company Ltd. (State Grid Electric Power Research Institute). He has served as the Director and a Secretary of the Party branch of East China Electric Power Dispatching and Communication

Center, the Deputy Chief Engineer of Zhejiang Electric Power Company, a Secretary of the Party Committee and the Deputy Director of Jinhua Electric Power Bureau, and the Deputy Director of the State Grid Marketing Department. He is the current Director of the China Electric Power Development Promotion Association, the Chairman of the China Electric Power Development Promotion Association Intelligent Power Industry Alliance, the Deputy Chairman of the National Professional Measurement Technology Committee, the Chairman of the Jiangsu Microcomputer Application Association, the Deputy Chairman of the Jiangsu Computer Society, and the Jiangsu Software Industry Association Vice Chairman. There are researches in the fields of electricity collection, electric vehicle charging and swapping technology, power dispatching, and telecommunications.



YANG LI received the Ph.D. degree in information number and information processing from the Nanjing University of Posts and Telecommunications, in 2010. He is currently a Senior Engineer with Nanjing Nanrui Information Communication Technology Company Ltd. His main research areas include power wireless communication technology and multimedia communication technology.



GAOFENG ZHAO (Member, IEEE) received the master's degree in power system and automation from the Electric Power Automation Research Institute of State Power Corporation, in April 1999. He was the Director of IEEE PES (China) Wired Communication Special Committee, and the Deputy Director of the Jiangsu Electrical Engineering Society Communication Special Committee. He is currently a Nanjing Nan Ruixin Deputy General Manager and a Senior Engineer of Tong Technology Company Ltd. His main research areas include power wireless communication private network planning and design technology, quantum communication technology, and SDN technology.



YIYANG NI received the B.S. degree in communications engineering and the Ph.D. degree from the Nanjing University of Posts and Telecommunications (NJUPT), Nanjing, China, in 2008 and 2016, respectively. She is currently doing the postdoctoral training in electronic science and technology. She is also an Associate Professor with Jiangsu Second Normal University and Jiangsu Institute of Educational Science Research. Her research interests include wireless communications, resource allocation in 5G, and the Internet of Things.



BO XU (Student Member, IEEE) received the B.S. degree from the Nanjing University of Posts and Telecommunications (NJUPT), Nanjing, China, in 2018, where he is currently pursuing the Ph.D. degree (successive postgraduate and doctoral programs) in communication and information systems. His research interests include mobile edge computing, big data, and distributed learning. He received the First-Class Scholarship and Special Freshman Scholarship from NJUPT, in 2018.



HAITAO ZHAO received the M.S. and Ph.D. (Hons.) degrees in signal and information processing from the Nanjing University of Posts and Telecommunications, Nanjing, China, in 2008 and 2011, respectively. He is currently an Associate Professor with the Nanjing University of Posts and Telecommunications. His current research interests include wireless multimedia modeling, capacity prediction, and wireless network coding.

...