Learning-Based Queue-Aware Task Offloading and Resource Allocation for Space–Air–Ground-Integrated Power IoT

Haijun Liao, *Graduate Student Member, IEEE*, Zhenyu Zhou, *Senior Member, IEEE*, Xiongwen Zhao, *Senior Member, IEEE*, and Yang Wang

***Abstract*—Space–air–ground-integrated power Internet of Things (SAG-PIoT) can provide ubiquitous communication and computing services for PIoT devices deployed in remote areas. In SAG-PIoT, the tasks can be either processed locally by PIoT devices, offloaded to edge servers through unmanned aerial vehicles (UAVs), or offloaded to cloud servers through satellites. However, the joint optimization of task offloading and computational resource allocation faces several challenges, such as incomplete information, dimensionality curse, and coupling between long-term constraints of queuing delay and short-term decision making. In this article, we propose a learning-based queue-aware task offloading and resource allocation algorithm (QUARTER). Specifically, the joint optimization problem is decomposed into three deterministic subproblems: 1) device-side task splitting and resource allocation; 2) task offloading; and 3) server-side resource allocation. The first subproblem is solved by the Lagrange dual decomposition. For the second subproblem, we propose a queue-aware actor–critic-based task offloading algorithm to cope with dimensionality curse. A greedy-based low-complexity algorithm is developed to solve the third subproblem. Compared with existing algorithms, simulation results demonstrate that QUARTER has superior performances in energy consumption, queuing delay, and convergence.**

***Index Terms*—Actor–critic, queue awareness, resource allocation, space–air–ground-integrated power Internet of Things (SAG-PIoT), task offloading.**

I. Introduction

With the rapid development of renewable energy, such as desert solar power and offshore wind power, a myriad of Power Internet-of-Things (PIoT) devices are deployed in remote areas to ensure the safe and reliable operation of power infrastructures [1]. PIoT is the smart grid-oriented Industrial IoT (IIoT), which poses stringent requirements on both communication and computational resources [2], [3]. However, these requirements can hardly be satisfied by the existing 5G network. On the one hand, these remote areas have little or even none 5G coverage [4]. It is predicted that above 80% ground area and above 95% ocean area still have no communication coverage in 5G. On the other hand, 5G network faces the problems of rigid resource deployment, poor flexibility, and weak emergence response capability [5], [6]. Therefore, it is imperative to develop an alternative network to provide seamless communication coverage and ubiquitous computing service in remote areas.

6G provides a feasible solution by leveraging space–air–ground-integrated network (SAGIN) to provide wide and flexible coverage [7]. The combination of SAGIN and PIoT, i.e., space–air–ground-integrated PIoT (SAG-PIoT) network, will shift PIoT from the 1-D ground networks to multidimensional heterogeneous networks. Specifically, SAG-PIoT consists of three segments, i.e., the space segment, aerial segment, and ground segment. The space segment includes the geostationary (GEO) satellites and low Earth orbit (LEO) satellites. GEO satellites provide wide-range communication coverage and act as relay nodes for the long-distance data transmission among LEO satellites, while LEO satellites provide comparatively high access rate and cloud computing functionality through satellite backbone network [8]. The aerial segment generally consists of high latitude platforms (HAPs), unmanned aerial vehicles (UAVs), and communication balloons, which can be flexibly deployed at locations with burst communication and computation demands to provide high-rate data transmission and multiconnection. Besides, edge servers can be employed to enhance on-demand edge computing capability and decrease transmission distance [9]–[11]. The ground segment is composed of massive PIoT devices with limited computation capability and battery capacity. The computation-intensive and delay-sensitive tasks generated by PIoT devices can be either processed locally, offloaded to the space segment, or offloaded to the aerial segment [12]. Meanwhile, the computational resources of devices, edge servers, and cloud servers should be jointly allocated with the task offloading decisions. However, the joint optimization of task offloading and resource allocation remains an open

Manuscript received October 1, 2020; revised December 29, 2020; accepted January 27, 2021. Date of publication February 9, 2021; date of current version March 24, 2021. This work was supported in part by the National Key Research and Development Program of China under Grant 2020YFB0905902; in part by the open research fund of National Mobile Communications Research Laboratory, Southeast University under Grant 2021D12; and in part by the Science and Technology Project of State Grid Corporation of China under Grant 5442GC200019. *(Corresponding author: Zhenyu Zhou.)*

Haijun Liao, Zhenyu Zhou, and Xiongwen Zhao are with Hebei Key Laboratory of Power Internet of Things Technology, North China Electric Power University, Beijing 102206, China, and also with National Mobile Communications Research Laboratory, Southeast University, Nanjing 211189, China (e-mail: haijun_liao@ncepu.edu.cn; zhenyu_zhou@ncepu.edu.cn; zhaoxw@ncepu.edu.cn).

Yang Wang is with the Institute of Information and Communication, China Electric Power Research Institute Co. Ltd, State Grid Corporation of China, Beijing 100192, China (e-mail: yangw@epri.sgcc.com.cn).

Digital Object Identifier 10.1109/JIOT.2021.3058236

issue. Several critical technical challenges are summarized as follows.

Coupling Between Task Offloading and Resource Allocation: The joint optimization problem is NP-hard due to the coupling among different entities and the coupling among different optimization variables in the same entity. Specifically, the optimization of resource allocation at edge servers and cloud servers directly affects the decisions of task offloading and resource allocation made by PIoT devices, and *vice versa*. Furthermore, for the same PIoT device, its task offloading and resource allocation decisions are also coupled.

Incomplete Information and Dimensionality Curse: It is impractical to know the perfect global state information (GSI), including channel state information (CSI), task arrival of PIoT devices, and computation capability of servers, due to the prohibitive signaling overhead and privacy concerns. Therefore, the conventional optimization approaches relying on prefect GSI are no longer suitable, and learning-based approaches should be employed. However, the numbers of system state and action grow exponentially due to the network heterogeneity and time-varying resource constraints, which invokes the problem of dimensionality curse. These approaches, which are not designed for high-dimensionality optimization problem, will have inferior performances in optimality and convergence.

Long-Term Constraints of Queuing Delay: Queuing delay conducts a major impact on the end-to-end delay performance and cannot be ignored in PIoT with stringent delay requirements. However, the consideration of queuing delay brings a new dimensionality of difficulty for joint optimization. Conventional learning-based approaches cannot be directly applied due to the coupling between the long-term constraints of queuing delay and the short-term decision making. Specifically, task offloading and resource allocation have to be jointly optimized without foreseeing the future states.

There exist some works on task offloading and resource allocation in SAGIN. Li *et al.* [13] proposed an iterative algorithm to jointly optimize channel resource allocation, uplink power control, and UAV relay deployment, the aim of which is to maximize the energy efficiency of the SAG-Internet of remote Things network. Shang and Liu [14] leveraged the coordinate descent algorithm and proposed an iterative resource allocation algorithm to minimize the energy consumption of mobile user equipments (UEs) in the air–ground-integrated mobile-edge computing networks by jointly optimizing UE association, channel allocation, transmission power, computation capacity, and UAV deployment. However, these works assume that perfect GSI is available, and cannot be applied for the scenario of incomplete GSI.

Reinforcement learning (RL) is originally developed to solve sequential decision-making problems with incomplete GSI. Sun *et al.* [15] developed an upper confidence bound (UCB)-based energy-aware mobility management (EMM) scheme to optimize delay performance under the long-term energy consumption constraint. In [16], a cooperative Q-learning-based task offloading scheme was proposed to learn the long-term optimal task offloading policy in a dynamic system environment. However, these works cannot efficiently handle the problems with large state space and action space since the high-dimensionality information has not been well exploited and learned [17]. Deep RL (DRL) provides a powerful methodology to optimize decision making under complex and dynamic network environment by reaping the learning and predicting capability provided by deep learning, and the decision-making capability provided by RL [18]. Cheng *et al.* [19] exploited policy gradient and actor–critic to learn the optimal offloading policy in SAGIN. Zhu *et al.* [20] proposed an advantage actor–critic RL approach to learn the near-optimal task migration policy in the UAV-enabled edge computing framework. However, these works lack queue awareness since the long-term constraints of queuing delay are ignored. Moreover, the device-side task splitting and resource allocation are also ignored.

Motivated by the aforementioned challenges, we propose a learning-based queue-aware task offloading and resource allocation algorithm (QUARTER). It can minimize the energy consumption of PIoT devices in the SAG-PIoT networks under the long-term constraints of queuing delay. First, Lyapunov optimization [21] is adopted to decouple the long-term constraints of queuing delay and short-term decision making. The formulated joint optimization problem is decomposed into three deterministic subproblems: 1) device-side task splitting and computational resource allocation; 2) task offloading; and 3) server-side computational resource allocation. The first subproblem is trivial to be solved by the Lagrange dual decomposition. Afterward, we formulate the second subproblem as a Markov decision process (MDP), and propose a queue-aware actor–critic-based task offloading algorithm (QAC) to cope with the problem of dimensionality curse, which can dynamically adjust task offloading strategies based on the queue backlog. Finally, we propose a greedy-based low-complexity algorithm to optimize server-side computational resource allocation, the principle of which is to preferentially allocate resources to the PIoT device with the largest target value.

The main contributions of this work are summarized as follows.

1) *Joint Optimization Problem Decomposition:* We leverage Lyapunov optimization to decompose the stochastic optimization problem into three deterministic subproblems, which are solved in a distributed and sequential manner.
2) *DRL-Based Task Offloading:* QUARTER can cope with the problem of dimensionality curse and dynamically learn the long-term optimal task offloading policy with incomplete GSI. It combines the complex function approximation capability provided by neural network and the decision-making capability provided by actor–critic.
3) *Queue Awareness:* QUARTER can achieve queue awareness by continuously adjusting task offloading and resource allocation strategies in accordance with queue information, which balances the tradeoff between energy consumption minimization and queuing delay reduction.

The remainder of this article is organized as follows. Section II introduces the system model. Section III presents the long-term queuing delay constraints, problem formulation, and Lyapunov optimization-based problem transformation. The

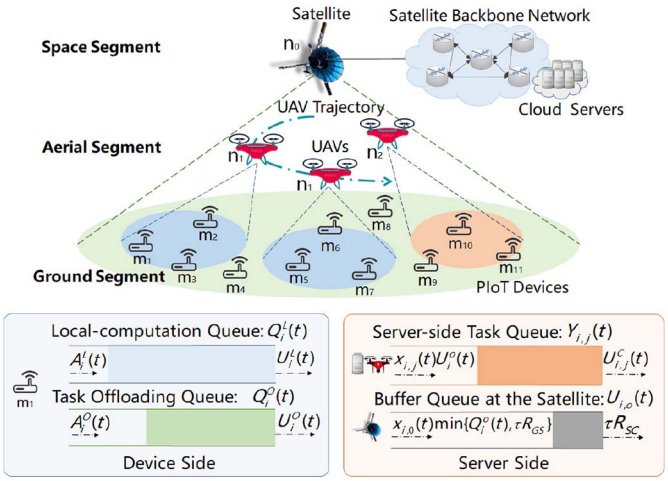

Fig. 1. System model.

optimization problem decomposition and the solution are provided in Section IV. The DRL-based task offloading algorithm is proposed in Section V. Section VI presents simulation results. Section VII concludes this article.

II. System Model

We consider an SAG-PIoT network, which is deployed in remote areas to provide seamless communication coverage and computing service to PIoT devices. As shown in Fig. 1, the SAG-PIoT network consists of three segments, i.e., the ground segment, aerial segment, and space segment. The ground segment is composed of PIoT devices with limited computation capability and battery capacity, the set of which is denoted as $\mathcal{M} = \{m_1, \ldots, m_i, \ldots, m_I\}$. In the aerial segment, J UAVs are equipped with base stations (BSs) and edge servers to provide network access and edge computing functionalities for ground PIoT devices [22]. In the space segment, an LEO satellite provides the full communication coverage for the area of interest. Since the PIoT devices are connected with the cloud servers through the satellite backbone network [19], cloud servers and satellite are denoted as the same symbol. Therefore, there are $J+1$ server candidates, the set of which is denoted as $\mathcal{N} = \{n_0, n_1, \ldots, n_j, \ldots, n_J\}$, where n_0 corresponds to the cloud server connected with the satellite, $n_1, \ldots, n_j, \ldots, n_J$ correspond to J edge servers connected with UAVs. The characteristics of UAVs and satellite are differentiated in three aspects, i.e., transmission rate, computational resource, and service availability. Specifically, UAVs can provide high data transmission rate and sufficient computational resources supported by the edge servers, but it is intermittently available for PIoT devices due to the mobility. Although satellite provides a comparatively smaller transmission rate, it possesses more powerful computational resources supported by the cloud servers and is always available for PIoT devices for a relatively long time due to the wide-range coverage.

A time-slotted model is adopted where the total optimization time period is divided into T time slots with equal duration τ, the set of which is denoted as $\mathcal{T} = \{1, \ldots, t, \ldots, T\}$ [23]. We consider a quasistatic scenario, where the environment, e.g., CSI and locations of UAVs, remains unchanged within one time slot but dynamically varies across different time slots [24]. In each time slot, tasks arrive at PIoT devices in a stochastic manner. We adopt the data-partition model in which each task can be divided into multiple subtasks with equal size A_0 (in bits) [25]. For PIoT device m_i, each arrived subtask is either computed locally or offloaded to a server. In the tth time slot, the amount of new tasks arriving at m_i in the tth time slot is denoted as $A_i(t)$, which can be split into two independent parts, i.e., $A_i^L(t)$ for local computation and $A_i^O(t)$ for task offloading. Thus, task splitting at m_i is given by

$$\begin{cases} A_i(t) = A_i^L(t) + A_i^O(t) \\ A_i^L(t), A_i^O(t) \in \{0, A_0, 2A_0, \ldots\}. \end{cases} \tag{1}$$

We assume m_i has two buffer queues to store tasks for local computation and task offloading, the backlogs of which are denoted as $Q_i^L(t)$ and $Q_i^O(t)$, respectively. $Q_i^L(t)$ and $Q_i^O(t)$ evolve as

$$Q_i^L(t+1) = \max\left\{Q_i^L(t) - U_i^L(t), 0\right\} + A_i^L(t) \tag{2}$$

$$Q_i^O(t+1) = \max\left\{Q_i^O(t) - U_i^O(t), 0\right\} + A_i^O(t) \tag{3}$$

where $U_i^L(t)$ and $U_i^O(t)$ represent the amounts of the task data leaving the device-side local-computation queue and the task offloading queue, respectively.

A. Task Offloading

Denote the server availability for m_i as a binary indicator $a_{i,j}(t) \in \{0, 1\}$, where $a_{i,j}(t) = 1$ represents that n_j is available for m_i in the tth time slot, i.e., m_i locates within the communication coverage of n_j, and $a_{i,j}(t) = 0$ otherwise. Denote task offloading strategy of m_i as a binary indicator $x_{i,j}(t) \in \{0, 1\}$, where $x_{i,j}(t) = 1$ represents that m_i selects n_j for task offloading in the tth time slot, and $x_{i,j}(t) = 0$ otherwise.

When tasks are offloaded from m_i to n_j, two different communication models are considered.

1) Ground-UAV Communication Model: For ground-UAV communication [20], the path loss between m_i and n_j is given by

$$L_{i,j,t} = 20\log_{10}\left(\frac{4\pi f_c\sqrt{d_{j,t}^2 + r_{i,j,t}^2}}{c}\right) + P_{i,j,t}^{\text{LoS}}\eta_{i,j,t}^{\text{LoS}} + \left(1 - P_{i,j,t}^{\text{LoS}}\right)\eta_{i,j,t}^{\text{NLoS}} \tag{4}$$

where $d_{j,t}$ and $r_{i,j,t}$ denote the flying altitude and the horizontal distance between m_i and n_j in the tth time slot, respectively. $\eta_{i,j,t}^{\text{LoS}}$ and $\eta_{i,j,t}^{\text{NLoS}}$ denote the additive loss incurred on top of the free space pathloss for Line-of-Sight (LoS) and Nonline-of-Sight (NLoS) links [26]. f_c denotes the carrier frequency, and c denotes the speed of light. $P_{i,j,t}^{\text{LoS}}$ is the LoS probability of ground-UAV link, which is given by

$$P_{i,j,t}^{\text{LoS}} = \frac{1}{1 + b_1\exp\left\{-b_2\left[\arctan\left(\frac{d_{j,t}}{r_{i,j,t}}\right) - b_1\right]\right\}}. \tag{5}$$

The values of b_1, b_2, $\eta_{i,j,t}^{\mathrm{LoS}}$, and $\eta_{i,j,t}^{\mathrm{NLoS}}$ are determined by the environment state. Thus, the transmission rate of ground-UAV link is calculated by

$$R_{i,j}(t) = B_{i,j}\log_2\left(1 + \frac{x_{i,j}(t)P_{\mathrm{TX}}10^{-L_{i,j,t}/10}}{\sigma^2}\right) \tag{6}$$

where P_{TX}, $B_{i,j}$, and σ^2 denote the transmission power, bandwidth, and noise power, respectively.

2) Ground-Satellite Communication Model: The satellite acts as the relay node between PIoT devices and cloud servers. According to [19], the ground-satellite transmission rate R_{GS} and the satellite-cloud transmission rate R_{SC} are assumed to be fixed, which are generally smaller than the ground-UAV transmission rate. The tasks stored at the satellite from m_i are modeled as a queue, the backlog of which is updated as

$$\begin{aligned} U_{i,0}(t+1) = &\max\left\{U_{i,0}(t) - \tau R_{\mathrm{SC}}, 0\right\} \\ &+ x_{i,0}(t)\min\left\{Q_i^O(t), \tau R_{\mathrm{GS}}\right\}. \end{aligned} \tag{7}$$

3) Throughput Model: The amount of tasks offloaded from m_i to n_j in the tth time slot is derived as

$$U_{i,j}^O(t) = \begin{cases} \min\left\{Q_i^O(t), \tau R_{i,j}(t)\right\}, & j = 1, \ldots, J \\ \min\left\{U_{i,0}(t), \tau R_{\mathrm{SC}}\right\}, & j = 0. \end{cases} \tag{8}$$

Thus, the amount of tasks leaving the device-side task offloading queue of m_i is given by

$$U_i^O(t) = \sum_{n_j \in \mathcal{N}} x_{i,j}(t)U_{i,j}^O(t). \tag{9}$$

B. Data Computation

We assume that n_j maintains a buffer queue to store the tasks offloaded from m_i. The backlog of server-side task queue is denoted as $Y_{i,j}(t)$ and updated as

$$Y_{i,j}(t+1) = \max\left\{Y_{i,j}(t) - U_{i,j}^C(t), 0\right\} + x_{i,j}(t)U_i^O(t) \tag{10}$$

where $U_{i,j}^C(t)$ is the amount of tasks offloaded from m_i and processed by n_j in the tth time slot.

1) Local Computation: Denote the amount of tasks that are processed by m_i locally in the tth time slot as $U_i^L(t)$, which is given by

$$U_i^L(t) = \min\left\{Q_i^L(t), \frac{\tau f_i(t)}{\lambda_i}\right\} \tag{11}$$

where $f_i(t)$ is the allocated CPU-cycle frequency of m_i, and λ_i is the computational density, i.e., the required CPU cycles per bit.

2) Computation at Remote Server: Denoting the allocated CPU-cycle frequency of n_j to process tasks offloaded from m_i in the tth time slot as $f_{i,j}(t)$, $U_{i,j}^C(t)$ is given by

$$U_{i,j}^C(t) = \min\left\{Y_{i,j}(t), \frac{\tau f_{i,j}(t)}{\lambda_i}\right\}. \tag{12}$$

C. Energy Consumption

The energy consumption of m_i consists of two parts, i.e., energy consumed for data computation $E_i^L(t)$ and that for task offloading $E_i^O(t)$. Specifically, $E_i^L(t)$ is given by

$$E_i^L(t) = \kappa\left[f_i(t)\right]^3 \min\left\{\frac{\lambda_i Q_i^L(t)}{f_i(t)}, \tau\right\} \tag{13}$$

where κ is the computation power parameter [27]. Given $x_{i,j}(t)$, $E_i^O(t)$ is derived as

$$\begin{aligned} E_i^O(t) = &x_{i,0}(t)P_{\mathrm{TX}}\min\left\{\frac{Q_i^O(t)}{R_{\mathrm{GS}}}, \tau\right\} \\ &+ \sum_{n_j \in \mathcal{N}\setminus n_0} x_{i,j}(t)a_{i,j}(t)P_{\mathrm{TX}}\min\left\{\frac{Q_i^O(t)}{R_{i,j}(t)}, \tau\right\} \end{aligned} \tag{14}$$

where the former and latter terms represent the energy consumption for task offloading through ground-satellite link and ground-UAV link, respectively. Therefore, the total energy consumption of m_i in the tth time slot is given by

$$E_i(t) = E_i^L(t) + E_i^O(t). \tag{15}$$

III. Problem Formulation and Transformation

In this section, we first introduce the queuing delay constraints and problem formulation. Then, the Lyapunov optimization-based problem transformation is elaborated.

A. Queuing Delay Constraints

Since queuing delay has a major impact on the end-to-end delay, we impose constraints on queuing delay to ensure the validity of the offloaded tasks. Specifically, the end-to-end queuing delay consists of three parts, i.e., queuing delays of transmission, computation, and result feedback.

1) Queuing Delay of Transmission: Based on *Little's Law* [28], the average queuing delay is proportional to the ratio of the average queue length to the average task arrival rate. Therefore, the queuing delay of transmission, i.e., queuing delay of $Q_i^O(t)$, is constrained as

$$\lim_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T}\frac{Q_i^O(t)}{\tilde{A}_i^O(t)} \le \tau_{i,\max}^{Q,O} \tag{16}$$

where $\tau_{i,\max}^{Q,O}$ is the queuing delay bound of transmission. $\tilde{A}_i^O(t)$ is the moving time-averaged task arrival rate of $Q_i^O(t)$, which is given by

$$\tilde{A}_i^O(t) = \frac{1}{t}\sum_{m=0}^{t-1}A_i^O(m). \tag{17}$$

2) Queuing Delay of Computation: The queuing delay of computation refers to the queuing delay of $Q_i^L(t)$ and that of $Y_{i,j}(t)$, which are similarly constrained as

$$\lim_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T}\frac{Q_i^L(t)}{\tilde{A}_i^L(t)}\le\tau_{i,\max}^{Q,L} \tag{18}$$

$$\lim_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T}\frac{Y_{i,j}(t)}{\tilde{U}_{i,j}^O(t)}\le\tau_{i,j,\max}^{Y} \tag{19}$$

where $\tau_{i,\max}^{Q,L}$ and $\tau_{i,j,\max}^{Y}$ are the corresponding queuing delay bounds of computation. $\tilde{A}_i^L(t)$ and $\tilde{U}_{i,j}^O(t)$ are given by

$$\tilde{A}_i^L(t)=\frac{1}{t}\sum_{m=0}^{t-1}A_i^L(m)$$

$$\tilde{U}_{i,j}^O(t)=\frac{1}{t}\sum_{m=0}^{t-1}x_{i,j}(m)U_{i,j}^O(m). \tag{20}$$

3) Queuing Delay of Result Feedback: Due to the high mobility, PIoT devices may locate outside the communication coverage of UAVs. Therefore, the results can be fed back only when the UAVs fly back, which results in the queuing delay of result feedback. To ensure the timeliness of results, the queuing delay of result feedback is constrained as

$$\lim_{T\to\infty}\frac{1}{T}\sum_{t=1}^{T}\big[1-a_{i,j}(t)\big]\mathbb{I}\big\{Y_{i,j}(t)f_{i,j}(t)\neq 0\big\}\tau\le\tau_{i,j,\max}^{Y,\text{out}} \tag{21}$$

where $\mathbb{I}\{x\}$ is an indicator function with $\mathbb{I}\{x\}=1$ if event x is true, and $\mathbb{I}\{x\}=0$ otherwise. $\tau_{i,j,\max}^{Y,\text{out}}$ is the queuing delay bound of result feedback.

B. Problem Formulation

The objective is to minimize the long-term time-averaged energy consumption of all the PIoT devices in the SAG-PIoT network by jointly optimizing task splitting, task offloading, and computational resource allocation. The joint optimization problem is formulated as

$$\begin{aligned}
\mathbf{P1}:\ &\min_{\mathbf{A^L}(t),\mathbf{A^O}(t),\mathbf{f}(t),\mathbf{x}(t)}\lim_{T\to\infty}\frac{1}{T}\sum_{t\in\mathcal{T}}\sum_{m_i\in\mathcal{M}}E_i(t)\\
\text{s.t.}\ \ &C_1: x_{i,j}(t)\in\{0,1\}\quad\forall m_i\in\mathcal{M}\quad\forall n_j\in\mathcal{N}\quad\forall t\in\mathcal{T}\\
&C_2:\sum_{n_j\in\mathcal{N}}x_{i,j}(t)=1\quad\forall m_i\in\mathcal{M}\quad\forall t\in\mathcal{T}\\
&C_3:\begin{cases}A_i(t)=A_i^L(t)+A_i^O(t)\\ A_i^L(t),A_i^O(t)\in\{0,A_0,2A_0,\ldots\}\end{cases}\\
&\qquad\forall m_i\in\mathcal{M}\quad\forall t\in\mathcal{T}\\
&C_4: f_i(t)\le f_{i,\max}(t)\quad\forall m_i\in\mathcal{M}\quad\forall t\in\mathcal{T}\\
&C_5:\sum_{m_i\in\mathcal{M}}f_{i,j}(t)\le f_{j,\max}(t)\quad\forall n_j\in\mathcal{N}\quad\forall t\in\mathcal{T}\\
&C_6: (18),\ (16),\ (19),\ \text{and}\ (21)\ \ \forall m_i\in\mathcal{M}\ \ \forall n_j\in\mathcal{N}.
\end{aligned} \tag{22}$$

$\mathbf{A^L}(t)=(A_i^L(t):m_i\in\mathcal{M})$ and $\mathbf{A^O}(t)=(A_i^O(t):m_i\in\mathcal{M})$ denote computation and task offloading, respectively. $\mathbf{f}(t)=(f_i(t),f_{i,j}(t):m_i\in\mathcal{M},\ n_j\in\mathcal{N})$ denotes the computational resource allocation vector. $\mathbf{x}(t)=(x_{i,j}(t):m_i\in\mathcal{M},\ n_j\in\mathcal{N})$ denotes the task offloading vector. C_1 and C_2 imply that each PIoT device can select only one server for task offloading in each time slot. C_3 specifies the task splitting constraints. C_4 and C_5 correspond to device-side and server-side computational resource allocation constraints, where $f_{i,\max}(t)$ and $f_{j,\max}(t)$ are the maximum available computational resources of m_i and n_j, respectively. C_6 denotes the queuing delay constraints.

C. Lyapunov Optimization-Based Problem Transformation

It is nontrivial to directly solve **P1** since the long-term queuing delay constraints are coupled with short-term decision making. Lyapunov optimization is leveraged to decouple the long-term stochastic optimization problem into a series of short-term deterministic optimization subproblems, which are solved in a slot-by-slot manner [21]. Based on the concept of virtual queue [29], the long-term queuing delay constraint C_6 can be converted to the queue stability constraint. Specifically, virtual queues $Z_i^{Q,L}(t)$, $Z_i^{Q,O}(t)$, $Z_{i,j}^{Y}(t)$, and $Z_{i,j}^{Y,\text{out}}(t)$ are introduced, which correspond to the constraints (16), (18), (19), and (21), respectively. The virtual queues evolve as

$$Z_i^{Q,L}(t+1)=\max\left\{Z_i^{Q,L}(t)+\frac{Q_i^L(t+1)}{\tilde{A}_i^L(t+1)}-\tau_{i,\max}^{Q,L},0\right\} \tag{23}$$

$$Z_i^{Q,O}(t+1)=\max\left\{Z_i^{Q,O}(t)+\frac{Q_i^O(t+1)}{\tilde{A}_i^O(t+1)}-\tau_{i,\max}^{Q,O},0\right\} \tag{24}$$

$$Z_{i,j}^{Y}(t+1)=\max\left\{Z_{i,j}^{Y}(t)+\frac{Y_{i,j}(t+1)}{\tilde{U}_{i,j}^O(t+1)}-\tau_{i,j,\max}^{Y},0\right\} \tag{25}$$

$$\begin{aligned}Z_{i,j}^{Y,\text{out}}(t+1)=\max\Big\{&Z_{i,j}^{Y,\text{out}}(t)-\tau_{i,j,\max}^{Y,\text{out}}+\big[1-a_{i,j}(t)\big]\mathbb{I}\\&\times\big\{Y_{i,j}(t)f_{i,j}(t)\neq 0\big\}\tau,0\Big\}.\end{aligned} \tag{26}$$

Theorem 1: If $Z_i^{Q,L}(t)$, $Z_i^{Q,O}(t)$, $Z_{i,j}^{Y}(t)$, and $Z_{i,j}^{Y,\text{out}}(t)$ are mean rate stable, C_6 holds automatically.

Proof: The detailed proof is omitted due to space limitation. A similar proof can be found in [21]. ■

Therefore, **P1** is equivalently transformed to

$$\begin{aligned}\mathbf{P2}:\ &\min_{\mathbf{A^L}(t),\mathbf{A^O}(t),\mathbf{f}(t),\mathbf{x}(t)}\lim_{T\to\infty}\frac{1}{T}\sum_{t\in\mathcal{T}}\sum_{m_i\in\mathcal{M}}E_i(t)\\
\text{s.t.}\ \ &C_1-C_5\\
&C_6^{'}: (23)-(26)\ \text{are mean rate stable.}\end{aligned} \tag{27}$$

To solve **P2**, for the sake of simplicity, we set $\Theta(t)=[Q_i^L(t),\ Q_i^O(t),\ Z_i^{Q,L}(t),\ Z_i^{Q,O}(t),\ Y_{i,j}(t),\ Z_{i,j}^{Y}(t),\ Z_{i,j}^{Y,\text{out}}(t)]$ to denote the concatenated vector of buffer queues and virtual queues. The Lyapunov function is defined as

$$\begin{aligned}L(\Theta(t))=\frac{1}{2}\Bigg\{\sum_{m_i\in\mathcal{M}}\Bigg[&Q_i^L(t)^2+Q_i^O(t)^2+Z_i^{Q,L}(t)^2+Z_i^{Q,O}(t)^2\\&+\sum_{n_j\in\mathcal{N}}\Big[Y_{i,j}(t)^2+Z_{i,j}^{Y}(t)^2+Z_{i,j}^{Y,\text{out}}(t)^2\Big]\Bigg]\Bigg\}.\end{aligned} \tag{28}$$

The one-step conditional Lyapunov drift is defined as the conditionally expected change of $L(\Theta(t))$ in two consecutive time slots, which is given by

$$\Delta L(\Theta(t)) = \mathbb{E}[L(\Theta(t+1)) - L(\Theta(t))|\Theta(t)]. \tag{29}$$

A positive/negative Lyapunov drift indicates that the queue backlogs become larger/smaller, and a smaller absolute value of Lyapunov drift indicates that the queue backlogs between two consecutive slots change slightly, which is essential to guarantee the queue stability.

We define the drift-plus-penalty to minimize the energy consumption under the constraints of queue stability, which is given by

$$\Delta_V L(\Theta(t)) = \Delta L(\Theta(t)) + V\mathbb{E}[\eta(t)|\Theta(t)] \tag{30}$$

where $\eta(t) = \sum_{m_i \in \mathcal{M}} E_i(t)$ denotes the total energy consumption of all the PIoT devices in the tth time slot. V is a nonnegative weight parameter, which trades off the "penalty minimization" and "queue stability," i.e., to minimize energy consumption and reduce queue length.

Theorem 2: Under all possible $\Theta(t)$ and $V \geq 0$, the drift-plus-penalty is upper bounded as

$$\begin{aligned}
&\Delta_V L(\Theta(t)) \\
&\leq C + \sum_{m_i \in \mathcal{M}} \mathbb{E}\left[Q_i^L(t)\left(A_i^L(t) - U_i^L(t)\right)|\Theta(t)\right] \\
&\quad + \sum_{m_i \in \mathcal{M}} \mathbb{E}\left[Q_i^O(t)\left(A_i^O(t) - U_i^O(t)\right)|\Theta(t)\right] \\
&\quad + \sum_{m_i \in \mathcal{M}} \mathbb{E}\left[Z_i^{Q,L}(t)\left(\frac{Q_i^L(t+1)}{\tilde{A}_i^L(t+1)} - \tau_{i,\max}^{Q,L}\right)|\Theta(t)\right] \\
&\quad + \sum_{m_i \in \mathcal{M}} \mathbb{E}\left[Z_i^{Q,O}(t)\left(\frac{Q_i^O(t+1)}{\tilde{A}_i^O(t+1)} - \tau_{i,\max}^{Q,O}\right)|\Theta(t)\right] \\
&\quad + \sum_{m_i \in \mathcal{M}} \sum_{n_j \in \mathcal{N}} \mathbb{E}\left[Y_{i,j}(t)\left(x_{i,j}(t)U_i^O(t) - U_{i,j}^C(t)\right)|\Theta(t)\right] \\
&\quad + \sum_{m_i \in \mathcal{M}} \sum_{n_j \in \mathcal{N}} \mathbb{E}\left[Z_{i,j}^Y(t)\left(\frac{Y_{i,j}(t+1)}{\tilde{U}_{i,j}^O(t+1)} - \tau_{i,j,\max}^Y\right)|\Theta(t)\right] \\
&\quad + \sum_{m_i \in \mathcal{M}} \sum_{n_j \in \mathcal{N}} \mathbb{E}\Big[Z_{i,j}^{Y,\text{out}}(t) \\
&\qquad \times \left(\left[1 - a_{i,j}(t)\right]\mathbb{I}\{Y_{i,j}(t)f_{i,j}(t) \neq 0\}\tau - \tau_{i,j,\max}^{Y,\text{out}}\right)|\Theta(t)\Big] \\
&\quad + V\mathbb{E}[\eta(t)|\Theta(t)]
\end{aligned} \tag{31}$$

where C is a positive constant, which does not affect the Lyapunov optimization.

Proof: The detailed proof is omitted due to space limitation. A similar proof can be found in [30]. ■

In (31), it is observed that task splitting, task offloading, and computational resource allocation are coupled. Denote $U_{i,j,\max}^O(t)$ as the maximum amount of tasks that can be offloaded from m_i to n_j in the tth time slot, which is referred as

$$U_{i,j,\max}^O(t) = x_{i,0}(t)\tau R_{\text{SC}} + \sum_{n_j \in \mathcal{N} \setminus n_0} \tau R_{i,j}(t). \tag{32}$$

By slightly loosing the upper bound, (31) can be further simplified and decoupled as the following theorem.

Theorem 3: Under all possible $\Theta(t)$ and $V \geq 0$, the upper bound of the drift-plus-penalty is transformed to

$$\begin{aligned}
&\Delta_V L(\Theta(t)) \\
&\leq C' + \sum_{m_i \in \mathcal{M}} \mathbb{E}\left[Q_i^L(t)\left(A_i^L(t) - U_i^L(t)\right)|\Theta(t)\right] \\
&\quad + \sum_{m_i \in \mathcal{M}} \mathbb{E}\left[Q_i^O(t)\left(A_i^O(t) - U_i^O(t)\right)|\Theta(t)\right] \\
&\quad + \sum_{m_i \in \mathcal{M}} \mathbb{E}\Bigg[Z_i^{Q,L}(t) \\
&\qquad \times \left(\frac{Q_i^L(t)}{\tilde{A}_i^L(t+1)} + \frac{A_i^L(t)}{\tilde{A}_i^L(t+1)} - \frac{U_i^L(t)}{\tilde{A}_i^L(t+1)}\right)|\Theta(t)\Bigg] \\
&\quad + \sum_{m_i \in \mathcal{M}} \mathbb{E}\Bigg[Z_i^{Q,O}(t)\Bigg(\frac{Q_i^O(t)}{\tilde{A}_i^O(t+1)} + \frac{A_i^O(t)}{\tilde{A}_i^O(t+1)} \\
&\qquad - \frac{(t+1)U_i^O(t)}{\sum_{m=0}^{t-1} A_i^O(m) + A_i(t)}\Bigg)|\Theta(t)\Bigg] \\
&\quad + \sum_{m_i \in \mathcal{M}} \sum_{n_j \in \mathcal{N}} \mathbb{E}\left[Y_{i,j}(t)\left(x_{i,j}(t)U_i^O(t) - U_{i,j}^C(t)\right)|\Theta(t)\right] \\
&\quad + \sum_{m_i \in \mathcal{M}} \sum_{n_j \in \mathcal{N}} \mathbb{E} \\
&\quad \times \Bigg[Z_{i,j}^Y(t)\Bigg(\frac{Y_{i,j}(t)}{\tilde{U}_{i,j}^O(t+1)} + \frac{x_{i,j}(t)U_i^O(t)}{\tilde{U}_{i,j}^O(t+1)} \\
&\qquad - \frac{(t+1)U_{i,j}^C(t)}{\sum_{m=0}^{t-1} x_{i,j}(m)U_{i,j}^O(m) + U_{i,j,\max}^O(t)}\Bigg)|\Theta(t)\Bigg] \\
&\quad + \sum_{m_i \in \mathcal{M}} \sum_{n_j \in \mathcal{N}} \mathbb{E}\Big[Z_{i,j}^{Y,\text{out}}(t) \\
&\qquad \times \left(\left[1 - a_{i,j}(t)\right]\mathbb{I}\{Y_{i,j}(t)f_{i,j}(t) \neq 0\}\tau\right)|\Theta(t)\Big] \\
&\quad + V\mathbb{E}[\eta(t)|\Theta(t)].
\end{aligned} \tag{33}$$

Proof: See the Appendix. ■

IV. Task Splitting and Resource Allocation

The solution to (33) can be obtained by minimizing the upper bound of the drift-plus-penalty in each time slot. Based on the principle of Lyapunov optimization, we decompose **P2** into three optimization subproblems **SP1**, **SP2**, and **SP3**, which can be solved sequentially and distributedly. The details and solutions are given as follows.

A. Joint Optimization of Task Splitting and Computational Resource Allocation at the PIoT Device Side

In **SP1**, m_i jointly determines the task splitting portion between local computation and task offloading, and the local CPU-cycle frequency allocation in the tth time slot. **SP1** is formulated as

$$\begin{aligned}\textbf{SP1}: \min_{A_i^L(t),A_i^O(t),f_i(t)} & VE_i^L(t)+Q_i^L(t)\left(A_i^L(t)-\frac{\tau f_i(t)}{\lambda_i}\right)\\ &+Q_i^O(t)A_i^O(t)\\ &+Z_i^{Q,L}(t)\left(\frac{Q_i^L(t)}{\tilde{A}_i^L(t+1)}+\frac{A_i^L(t)}{\tilde{A}_i^L(t+1)}-\frac{\tau f_i(t)}{\lambda_i\tilde{A}_i^L(t+1)}\right)\\ &+Z_i^{Q,O}(t)\left(\frac{Q_i^O(t)}{\tilde{A}_i^O(t+1)}+\frac{A_i^O(t)}{\tilde{A}_i^O(t+1)}\right)\\ \text{s.t. } & C_3 \text{ and } C_4\\ & C_7: \frac{\tau f_i(t)}{\lambda_i}\le Q_i^L(t). \end{aligned} \tag{34}$$

Plugging C_3 and (13) into (34), **SP1** is simplified as

$$\begin{aligned}\textbf{SP1}': \min_{A_i^L(t),f_i(t)} & \Phi\big(A_i^L(t),f_i(t)\big)=V\kappa\big[f_i(t)\big]^3\min\left\{\frac{\lambda_i Q_i^L(t)}{f_i(t)},\tau\right\}\\ &+Q_i^L(t)\left(A_i^L(t)-\frac{\tau f_i(t)}{\lambda_i}\right)\\ &+Q_i^O(t)\big[A_i(t)-A_i^L(t)\big]\\ &+\frac{(t+1)Z_i^{Q,L}(t)}{\lambda_i}\left(\frac{\lambda_i Q_i^L(t)+\lambda_i A_i^L(t)-\tau f_i(t)}{A_i^L(t)+\sum_{m=0}^{t-1}A_i^L(m)}\right)\\ &+(t+1)Z_i^{Q,O}(t)\left(\frac{Q_i^O(t)+A_i(t)-A_i^L(t)}{A_i(t)-A_i^L(t)+\sum_{m=0}^{t-1}A_i^O(m)}\right)\\ \text{s.t. } & C_4 \text{ and } C_7\\ & C_8: A_i^L(t),A_i^O(t)\in\{0,A_0,2A_0,\ldots\}. \end{aligned} \tag{35}$$

SP1′ is a convex optimization problem and can be solved by the Lagrange dual decomposition [31]. Denote ξ_i and ρ_i as the vectors of Lagrange multipliers corresponding to constraints C_4 and C_7. The Lagrangian function associated with **SP1**′ is given by

$$\begin{aligned}\mathcal{L}\big(A_i^L(t),f_i(t),\xi_i,\rho_i\big)=&\Phi\big(A_i^L(t),f_i(t)\big)+\xi_i\big[f_i(t)-f_{i,\max}(t)\big]\\ &+\rho_i\left[\frac{\tau f_i(t)}{\lambda_i}-Q_i^L(t)\right].\end{aligned} \tag{36}$$

Based on the Lagrange dual decomposition, (36) can be decomposed as

$$\max_{(\xi_i(t)>0,\rho_i(t)>0)}\ \min_{(A_i^L(t),f_i(t))}\ \mathcal{L}\big(A_i^L(t),f_i(t),\xi_i(t),\rho_i(t)\big). \tag{37}$$

Then, by employing the Karush–Kuhn–Tucker conditions, the optimal values of $A_i^L(t,z+1)$ and $f_i(t,z+1)$ are given by

$$A_i^L(t,z+1)=\max\Big\{\arg_{A_i^L(t)}\ \nabla_{A_i^L(t)}\mathcal{L}=0,0\Big\} \tag{38}$$

$$f_i(t,z+1)=\max\big\{\arg_{f_i(t)}\ \nabla_{f_i(t)}\mathcal{L}=0,0\big\} \tag{39}$$

where z is the iteration index of Lagrange multiplier updating. $\nabla_{A_i^L(t)}\mathcal{L}=0$ is a quartic equation in one variable, which can be solved by Ferrari's method [32]. Due to space limitation, the details are omitted here. $\nabla_{f_i(t)}\mathcal{L}=0$ is a quadratic equation in one variable and can be solved easily.

By exploiting the gradient method [33], the Lagrange multipliers are updated as

$$\xi_i(t,z+1)=\max\big\{\xi_i(t,z)+\big[f_i(t,z)-f_{i,\max}(t)\big]\psi_{\xi_i}(t,z),0\big\} \tag{40}$$

$$\rho_i(t,z+1)=\max\left\{\rho_i(t,z)+\left[\frac{\tau f_i(t)}{\lambda_i}-Q_i^L(t)\right]\psi_{\rho_i}(t,z),0\right\} \tag{41}$$

where $\psi_{\xi_i}(t,z)$ and $\psi_{\rho_i}(t,z)$ are the step sizes, which should be carefully determined to tradeoff the performances of convergence and optimality.

B. Task Offloading Optimization

In **SP2**, m_i determines either to offload tasks to the satellite n_0 or one of the UAVs, e.g., $n_j\in\mathcal{N}\setminus n_0$. In this regard, **SP2** is formulated as

$$\begin{aligned}\textbf{SP2}: \min_{\{x_{i,j}(t)\}} & \Gamma\big(x_{i,j}(t)\big)\\ &=VE_i^O(t)+\sum_{n_j\in\mathcal{N}}Y_{i,j}(t)x_{i,j}(t)U_i^O(t)\\ &\quad-\frac{(t+1)Z_i^{Q,O}(t)U_i^O(t)}{\sum_{m=0}^{t-1}A_i^O(m)+A_i(t)}-Q_i^O(t)U_i^O(t)\\ &\quad+\sum_{n_j\in\mathcal{N}}\left[Z_{i,j}^Y(t)\left(\frac{Y_{i,j}(t)}{\tilde{U}_{i,j}^O(t+1)}+\frac{x_{i,j}(t)U_i^O(t)}{\tilde{U}_{i,j}^O(t+1)}\right)\right]\\ \text{s.t. } & C_1\sim C_2. \end{aligned} \tag{42}$$

It is nontrivial to directly solve **SP2** due to the lack of complete GSI. We formulate the task offloading problem in SAG-PIoT as an MDP and leverage DRL to cope with the problem of dimensionality curse. Details are elaborated in Section V.

C. Server-Side Resource Allocation

In **SP3**, n_j determines the amount of allocated CPU-cycle frequency to process tasks offloaded from m_i in the tth time slot. **SP3** is formulated as

$$\begin{aligned}\textbf{SP3}: \max_{\{f_{i,j}(t)\}} & \Psi(f_{i,j}(t))=\sum_{m_i\in\mathcal{M}}\frac{\tau Y_{i,j}(t)f_{i,j}(t)}{\lambda_i}\\ &+\sum_{m_i\in\mathcal{M}}\frac{\tau(t+1)f_{i,j}(t)}{\lambda_i\Big(\sum_{m=0}^{t-1}x_{i,j}(m)U_{i,j}^O(m)+U_{i,j,\max}^O(t)\Big)}\\ &-\sum_{m_i\in\mathcal{M}}\Big[Z_{i,j}^{Y,\text{out}}(t)\big(\big[1-a_{i,j}(t)\big]\mathbb{I}\\ &\qquad\times\big\{Y_{i,j}(t)f_{i,j}(t)\neq 0\big\}\tau\big)\Big]\\ \text{s.t. } & C_5\\ & C_9: \frac{\tau f_{i,j}(t)}{\lambda_i}\le Y_{i,j}(t)\quad\forall m_i\in\mathcal{M}. \end{aligned} \tag{43}$$

The main concept of the server-side greedy-based computational resource allocation algorithm is to preferentially allocate resources to the PIoT device, which can achieve larger $\Psi(f_{i,j}(t))$. Details are given in Algorithm 1. The main concept is, first initialize the set of PIoT devices with computational resource requirements as $\mathcal{M}_t=\{m_i\in\mathcal{M}|Y_{i,j}(t)>0\}$ and the available computational resources as $\Delta f_{j,\max}(t)=$

Algorithm 1 Server-Side Computational Resource Allocation

1: Input: $\{Y_{i,j}(t)\}$, $\{\lambda_i\}$, $\{\sum_{m=0}^{t-1} x_{i,j}(m)U_{i,j}^{O}(m)\}$, $\{x_{i,j}(t)\}$, $\{U_{i,j,max}^{O}(t)\}$, $\{a_{i,j}(t)\}$, and $\{Z_{i,j}^{Y,out}(t)\}$.
2: **Initialize** $\mathcal{M}_t = \{m_i \in \mathcal{M} | Y_{i,j}(t) > 0\}$, and $\Delta f_{j,max}(t) = f_{j,max}(t)$.
3: **while** $\mathcal{M}_t \neq \emptyset$ and $\Delta f_{j,max}(t) > 0$ **do**
4: $\varphi_{i,j}(t) = \min\{\Delta f_{j,max}(t), \frac{\lambda_i}{\tau}\big[Y_{i,j}(t)\big]\}$.
5: $m_{i^*} = \arg\max_{m_i \in \mathcal{M}_t} \Psi(\varphi_{i,j}(t))$.
6: $f_{i^*,j}(t) = \varphi_{i^*,j}(t)$.
7: $\Delta f_{j,max}(t) = \Delta f_{j,max}(t) - f_{i^*,j}(t)$.
8: $\mathcal{M}_t = \mathcal{M}_t \setminus m_{i^*}$.
9: **end while**.

$f_{j,\max}(t)$. Then, calculate the maximum amount of computational resources allocated to m_i as $\varphi_{i,j}(t)$ and the target value $\Psi(\varphi_{i,j}(t))$. The greedy one, i.e., the PIoT device m_{i^*} with the maximum target value $\Psi(\varphi_{i,j}(t))$ will be selected. Afterward, remove m_{i^*} from the set $\mathcal{M}_t$ and update $\Delta f_{j,\max}(t)$. The iterations of the computational resource allocation terminate when there are no PIoT devices on demand, i.e., $\mathcal{M}_t = \emptyset$, or no excess computational resources are available, i.e., $\Delta f_{j,\max}(t) = 0$.

V. DRL-Based Queue-Aware Task Offloading Decision Making

In this section, we first introduce the MDP-based network environment. Next, two kinds of RL-based task offloading methods are demonstrated. Finally, we propose a queue-aware actor–critic-based task offloading algorithm named QAC to solve **SP2**.

A. MDP-Based Network Environment

The task offloading problem in SAG-PIoT, i.e., **SP2**, can be modeled as an MDP, which is defined as a tuple (S, A, T, R). S is the system state set, A is the action set, T is the transition probability set, and R is the immediate reward/cost function with respect to state $s \in S$ and action $a \in A$. Policy π is the mapping from S to A. The MDP of the SAG-PIoT task offloading problem is modeled as follows.

1) State: The system state includes queue backlog, task-related information, and empirical network performance, which is denoted as $\mathbf{S}_i(t) = \{\mathbf{Y}_{i,j}(t), \mathbf{Z}_{i,j}^{Y}(t), Z_i^{Q,O}(t), Q_i^{O}(t), \sum_{m=0}^{t-1} A_i^{O}(m), A_i(t), \tilde{\mathbf{U}}_{i,j}^{O}(t)\}$. The state information of the previous $(t-1)$ time slots, i.e., $\mathbf{S}_i(1), \ldots, \mathbf{S}_i(t-1)$, and that of the current time slot, i.e., $\mathbf{S}_i(t)$, are utilized to learn and predict the future system states.

2) Action: At the beginning of the tth time slot, each PIoT device, e.g., m_i, takes the action of task offloading, i.e., to determine $x_{i,j}(t) \quad \forall n_j \in \mathcal{N}$, the set of which is denoted as $\mathbf{A}_i(t) = \{x_{i,0}(t), x_{i,1}(t), \ldots, x_{i,J}(t)\}$.

3) Transition Probability: Since the task arrival is irrelevant to the task offloading action, the system transition probability is calculated by

$$\begin{aligned} &P(\mathbf{S}_i(t+1)|\mathbf{S}_i(t), \mathbf{A}_i(t)) \\ &\quad = P\big(\mathbf{Y}_{i,j}(t+1)|\mathbf{Y}_{i,j}(t), \mathbf{A}_i(t)\big) \\ &\qquad \times P\Big(\mathbf{Z}_{i,j}^{Y}(t+1)|\mathbf{Z}_{i,j}^{Y}(t), \mathbf{A}_i(t)\Big) \\ &\qquad \times P\Big(Z_i^{Q,O}(t+1)|Z_i^{Q,O}(t), \mathbf{A}_i(t)\Big) \\ &\qquad \times P\Big(Q_i^{O}(t+1)|Q_i^{O}(t), \mathbf{A}_i(t)\Big) \\ &\qquad \times P\Big(\tilde{\mathbf{U}}_{i,j}^{O}(t+1)|\tilde{\mathbf{U}}_{i,j}^{O}(t), \mathbf{A}_i(t)\Big). \end{aligned} \tag{44}$$

The numbers of system state and action grow exponentially as the number of servers increases, which makes it intractable to accurately model the transition probability.

4) Reward: Since **SP2** is a minimization problem, cost function is employed, which is defined as the optimization objective in **SP2**, i.e., $\Gamma(x_{i,j}(t))$.

B. RL-Based Task Offloading

To cope with the modeling difficulty of transition probability, we resort to two model-free RL-based methods, i.e., value-based methods represented by Q-learning [34] and policy-based methods represented by policy gradient method [17].

1) Value-Based Methods: Value-based methods have great potential in optimizing the deterministic policy of RL problems with large state space. Define the value function of state s as the expected long-term discounted reward with policy π, i.e.,

$$V(s|\pi) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t \Gamma(s_t, a_t)|s_0 = s, \pi\right] \tag{45}$$

where $\gamma \in [0, 1]$ is the discount factor. The optimal policy $\pi^*(s)$ is obtained by minimizing the value function of each state, which is given by

$$\pi^*(s) = \arg\min_{a} \sum_{s'} P\big(s'|s, a\big)\big[\Gamma(s, a) + \gamma V\big(s'|\pi^*\big)\big] \tag{46}$$

where $P(s'|s, a)$ is the stationary distribution of Markov chain generated by $\pi^*(s)$.

Q-learning estimates the action-value function starting from state s with action a as $Q(s, a)$, which is given by

$$Q(s, a) = Q(s, a) + \alpha\Big(R + \gamma \max_{a} Q(s', a) - Q(s, a)\Big) \tag{47}$$

where α is the learning rate. In each time slot, Q-learning leverages ϵ-greedy algorithm to select an action based on the estimated $Q(s, a)$. As the system state space and action space become larger, it takes enormous storage resources to store $Q(s, a)$, which lowers the practicability and feasibility. One promising solution is to introduce a parameter ω to approximate the value function as

$$\hat{Q}(s, a, \omega) \approx Q_{\pi}(s, a). \tag{48}$$

ω can be progressively approximated through linear function, nearest neighbor, and neural network.

2) Policy-Based Methods: Compared with value-based methods, policy-based methods are more suitable for optimizing stochastic policy. The optimal policy can be obtained by employing the conventional gradient ascent/descent update method, which possesses powerful and appealing nonlinear parameter approximation functionality and can effectively deal with the large and continuous action spaces. A parameter θ is introduced to parameterize and approximate the policy as

$$\pi_\theta(s,a) = P(a|s,\theta) \approx \pi(a|s) \tag{49}$$

where $\pi_\theta(s,a)$ represents the probability that action a is drawn in state s under the policy π with parameter θ. The expected value function of the start state is defined as

$$J(\theta) = V_{\pi_\theta}(s_0) = \mathbb{E}\left[\sum_{t=0}^{T} \gamma^t \Gamma(s_t,a_t)|\pi_\theta(s_t,a_t)\right]. \tag{50}$$

For the cost minimization problem, the gradient descent method is adopted to progressively update the policy approximation parameter θ along the direction of cost reduction [35]. The update process of θ is given by

$$\theta(t+1) = \theta(t) - \psi \nabla J(\theta(t)) \tag{51}$$

where ψ is the learning rate, and $\nabla J(\theta(t))$ is derived based on the policy gradient theorem [36] as

$$\nabla J(\theta(t)) = \mathbb{E}_\pi\left[\nabla_\theta \log \pi(s_t,a_t) Q_\pi(s_t,a_t)\right]. \tag{52}$$

C. Queue-Aware Actor–Critic-Based Task Offloading

The proposed QAC algorithm incorporates the potential of value-based methods in solving the problem with large state space and the capability of policy-based methods in learning the stochastic policy. It consists of an actor network and a critic network by approximating both the value function and the policy simultaneously. The actor network optimizes the policy, draws actions, and interacts with the environment based on the policy-based method while the critic network generates the value function to evaluate and criticize the current policy and guides policy updating. The environment of QAC refers to all the network states that are outside the PIoT devices and cannot be changed arbitrarily by PIoT devices [17], such as CSI, available computational resources of servers, server availability, and so on. Furthermore, neural network is adopted to learn the parameters ω and θ due to the powerful capability to approximate complex functions. The specific implementation of QAC is summarized in Algorithm 2.

The queue information, task-related information, and empirical network performance are input to both the actor network and the critic network. We assume a total of G episodes, each of which consists of T time slots. Considering the prohibitive switching cost between the ground-UAV link and the ground-satellite link in the form of delay, the task offloading decision is optimized every D time slots. For the current time slot $t = kD+1, k = 0,1,2,\ldots,$ the actor network first draws action $a_i(t)$ based on the policy $\pi(\mathbf{S}_i(t)|\theta)$. Otherwise, the task offloading action remains unchanged, i.e., $a_i(t) = a_i(t-1)$. It is noted that $\mathbf{S}_i(t)$ is the state forwarded from the environment to the PIoT devices. Second, m_i executes action $a_i(t)$. Assuming that $a_i(t) = j$, which is equivalent to $x_{i,j}(t) = 1$, m_i offloads the tasks to the server n_j. Third, m_i observes $U^O_{i,j}(t)$ and infers $f_{i,j}(t)$ based on the feedback results. Finally, m_i calculates the cost $\Gamma(s_i(t),a_i(t))$ based on the state $s_i(t)$ and action $a_i(t)$, and the system transfers to the next state, i.e., $s_i(t+1)$.

Algorithm 2 QAC Task Offloading Algorithm

1: **Input**: Queue information: $\mathbf{Y}_{i,j}(t)$, $Q^O_i(t)$, $\mathbf{Z}^Y_{i,j}(t)$, $Z^{Q,O}_i(t)$
2: Task-related information: $\sum_{m=0}^{t-1} A^O_i(m)$, $A_i(t)$
3: Empirical network performance: $\tilde{\mathbf{U}}^O_{i,j}(t)$
4: **Output**: Task offloading decision $\{x_{i,j}(t)\}$
5: **For** episode $e = 1,\cdots,G$ **do**
6: **For** time slot $t = 1,\cdots,T$ **do**
7: **If** $t = kD+1, k = 0,1,2\cdots,$ **do**
8: Draw action $a_i(t)$ for **SP2** based on the policy $\pi(\mathbf{S}_i(t)|\theta)$.
9: **else**
10: $a_i(t)$=$a_i(t-1)$.
11: **end if**
12: Execute action $a_i(t)$.
13: Observe $U^O_{i,j}(t)$ and infer $f_{i,j}(t)$.
14: Calculate cost $\Gamma(s_i(t),a_i(t))$ and transfer to the next state $s_i(t+1)$.
15: Calculate $\phi = \Gamma(s_i(t),a_i(t)) + \gamma V(s_i(t+1),\omega) - V(s_i(t),\omega)$.
16: Update $\omega = \omega - \psi' \nabla_\omega \phi^2$.
17: Update $\theta = \theta - \psi \phi \nabla_\theta \log \pi(s_i(t),a_i(t)|\theta)$.
18: Update $Q^O_i(t)$, $\mathbf{Y}_{i,j}(t)$, $Z^{Q,O}_i(t)$, and $\mathbf{Z}^Y_{i,j}(t)$ as (3), (10), (24), and (25).
19: **end for**
20: **end for**

Compared with the conventional network update methods based on discounted return of cost $G_t = \Gamma(t) + \gamma\Gamma(t+1) + \cdots + \gamma^{T-t}\Gamma(T)$ in each episode, we leverage the temporal-difference (TD) error to update networks in each time slot, which can substantially accelerate the convergence speed. Define the TD error as $\phi = \Gamma(s_i(t),a_i(t)) + \gamma V(s_i(t+1),\omega) - V(s_i(t),\omega)$. The update processes of the critic network parameter ω and the actor network parameter θ are shown in lines 16 and 17. ψ and ψ' are the learning rates for the actor network and the critic network, respectively. Next, update the queue information $Q^O_i(t)$, $\mathbf{Y}_{i,j}(t)$, $Z^{Q,O}_i(t)$, and $\mathbf{Z}^Y_{i,j}(t)$ as (3), (10), (24), and (25). The iteration stops until $e > G$.

As the core of QUARTER, QAC achieves queue awareness, i.e., it can dynamically adjust the task offloading strategies based on the queue information. Specifically, when the queue backlog increases severely and the queuing delay deviates from the corresponding requirement, the cost and TD error become larger, which enforces m_i to update the actor network as well as the critic network, optimize task offloading policy, and select another server with superior queuing delay performance, thereby enabling queue awareness.

TABLE I
SIMULATION PARAMETERS

Parameter	Value	Parameter	Value
T	100	τ	100 ms
I	1000	J	3
$\eta_{i,j,t}^{LoS}, \eta_{i,j,t}^{NLoS}$	0.1, 21	P_{TX}	23 dBm
f_c	0.1 GHz	δ^2	-114 dBm
$d_{j,t}$	90 m	b_1, b_2	4.88, 0.43
γ	0.99	λ	1000 CPU cycles/bit
$\tau_{i,max}^{Q,L}$	70 ms	$\tau_{i,max}^{Q,O}$	50 ms
$\tau_{i,j,max}^{Y}$	50 ms	$\tau_{i,j,max}^{Y,out}$	50 ms
V	10^6	κ	5×10^{-26} J/Hz^3/s
G	10	$B_{i,j}$	1 MHz

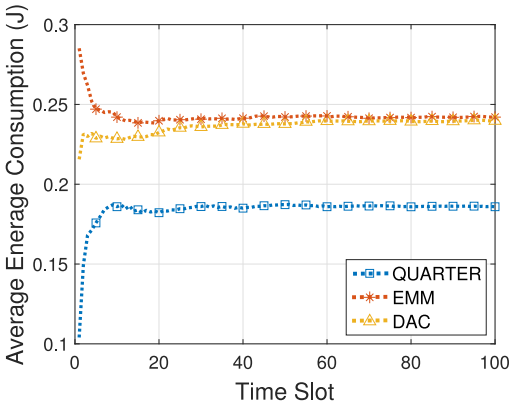

Fig. 2. Energy consumption performance.

VI. PERFORMANCE ANALYSIS AND SIMULATIONS

In this section, we evaluate the proposed QUARTER through simulations. The simulation is performed via MATLAB and run over ThinkStation P520 with Intel Core i7-6900K CPU and 48-GB random access memory (RAM). We consider a remote 1000 m $\times$ 1000 m square area with 1000 PIoT devices, 3 UAVs, and 1 LEO satellite. The locations of PIoT devices are randomly distributed within the considered area. The UAVs fly in circle with the same center and a radius of 200 m, and the angle between any two UAVs is 120°. The UAVs' flying altitude and communication coverage radius are set as 90 and 300 m. The number of arrived task $A_i(t)$ follows a uniform distribution within the interval $[240, 360] \times 10^4$ bits, and the subtask size is set as $A_0 = 10^4$ bits. The maximum available computational resources of PIoT devices, cloud servers, and edge servers, i.e., $f_{i,\max}(t)$, $f_{0,\max}(t)$, and $f_{j,\max}(t)$, $j = 1, \ldots, J$, are uniformly distributed within the interval [1.6, 2.4], [264, 396], and [79.2, 118.8] GHz, respectively. The ground-satellite transmission rate R_{GS} and the satellite-cloud transmission rate R_{SC} are both set as 20 Mb/s. The other detailed simulation parameters are shown in Table I due to space limitation [25]. Two state-of-the-art algorithms are leveraged as comparison. The first one is the EMM [15] based on UCB, in which the energy deficit is replaced by the queuing delay deficit. The second one is the deep actor–critic-based online computing offloading algorithm (DAC) [19], which ignores queue awareness. In EMM and DAC, the task splitting portion and local-computation CPU-cycle frequency remain fixed, and servers evenly allocate computational resources.

Fig. 2 shows the average energy consumption of PIoT devices versus time slots. The task splitting portions of EMM and DAC are both set as 0.8, which means that 80% of tasks are processed locally. When $t = 100$, QUARTER outperforms DAC and EMM by 22.36% and 23.13%, respectively. QUARTER performs the best since it jointly optimizes device-side task splitting and resource allocation, thereby resulting in less local-computation energy consumption that plays a major role in reducing energy consumption. The energy consumption of DAC and EMM is basically the same due to the fixed task splitting portion. It validates the effectiveness of exploiting dynamic task splitting adjustment to reduce energy consumption.

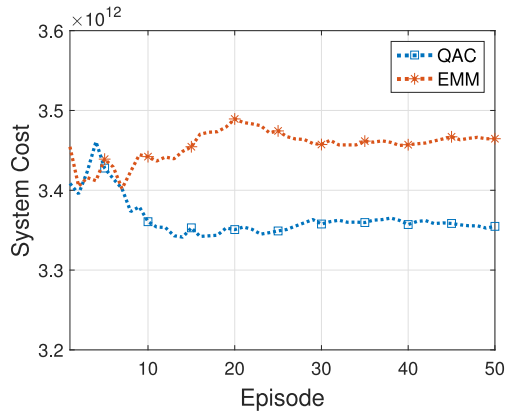

Fig. 3. Convergence performance.

Fig. 3 shows the convergence performance of QAC. System cost is defined as the sum of the cost $\Gamma(x_{i,j}(t))$. It can be seen that QAC converges at the 15th episode. The superior convergence speed stems from the TD error-based network update in each time slot rather than in each episode, and the capability to approximate complex function provided by neural network.

Fig. 4(a)–(c) shows the average queue backlogs of Q_i^L, Q_i^O, and $Y_{i,j}$ versus time slots, respectively. Compared with QUARTER and DAC, EMM has larger average backlog of Q_i^O since UCB cannot handle the high-dimensionality optimization problem with large state space and action space, which results in inferior task offloading performance and fewer tasks offloaded from Q_i^O to the servers. Both EMM and DAC perform worse than QUARTER in the average backlog of Q_i^L and $Y_{i,j}$ due to the queue unawareness and fixed computational resource allocation strategy.

Fig. 5(a)–(c) shows the average queuing delays of Q_i^L, Q_i^O, and $Y_{i,j}$ versus time slots, respectively. Numerical results indicate that compared with DAC, QUARTER reduces queuing delay of Q_i^L, Q_i^O, and $Y_{i,j}$ by 44.38%, 8.52%, and 59.03%. QUARTER has superior queuing delay performances through reaping the benefits of both the queue awareness and the joint optimization of task offloading and resource allocation.

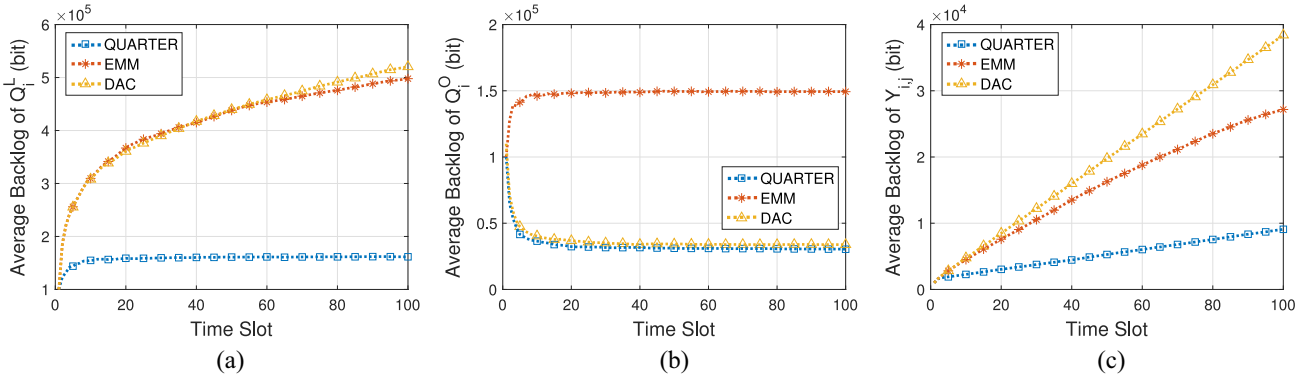

Fig. 4. Average queue backlog performances. (a) Average backlog of local-computation queue Q_i^L. (b) Average backlog of device-side task offloading queue Q_i^O. (c) Average backlog of server-side task queue $Y_{i,j}$.

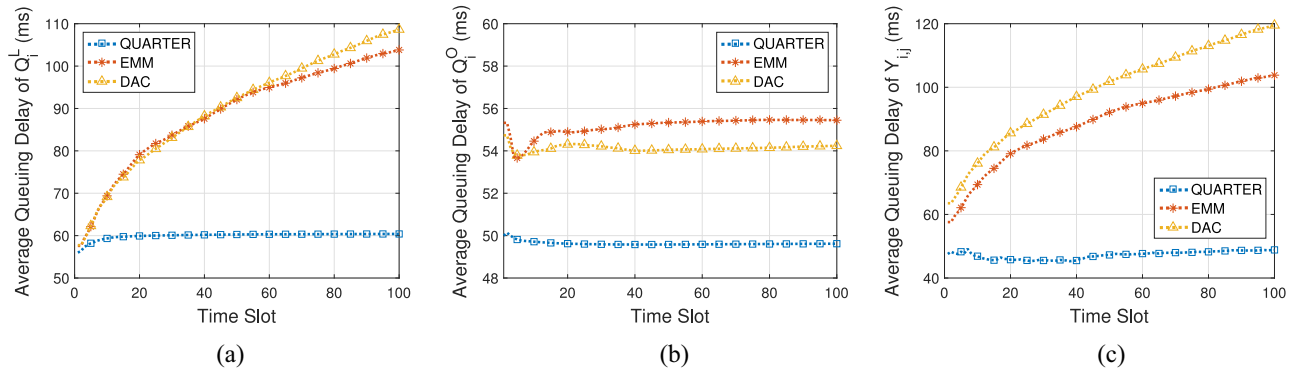

Fig. 5. Average queuing delay performances. (a) Average queuing delay of Q_i^L. (b) Average queuing delay of Q_i^O. (c) Average queuing delay of $Y_{i,j}$.

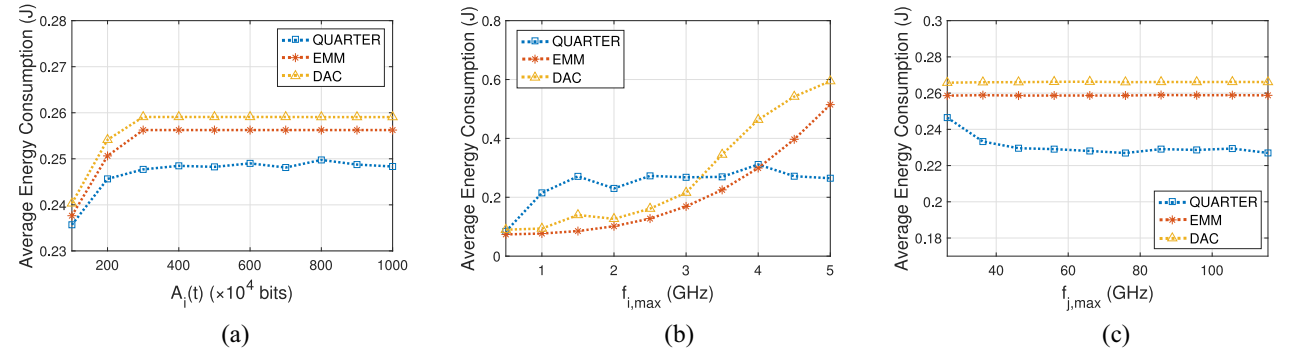

Fig. 6. Impacts of system parameters on the energy consumption. (a) Impact of task arrival $A_i(t)$. (b) Impact of device-side computing capability $f_{i,\max}(t)$. (c) Impact of server-side computing capability $f_{j,\max}(t)$.

Fig. 6(a)–(c) shows the impacts of task arrival $A_i(t)$, device-side computing capability $f_{i,\max}(t)$, and server-side computing capability $f_{j,\max}(t)$ on the average energy consumption. In Fig. 6(a), as task arrival increases, energy consumption increases first and then flattens. The reason is that more energy is consumed for local task processing and task offloading as task arrival increases. However, when task arrival becomes comparatively large, the limited computing and communication capabilities are the bottlenecks for task processing and offloading, thereby energy consumption remaining unchanged. Comparing the energy consumption increment when $A_i(t)$ increases from 10^6 to 3×10^6 bits, QUARTER outperforms EMM and DAC by 35.48% and 36.17% due to the endowed queue awareness and the joint optimziation of task offloading and resource allocation.

As shown in Fig. 6(b), the energy consumption of EMM and DAC grows substantially while QUARTER maintains energy consumption in a comparatively low and small fluctuation level. The reason is that QUARTER dynamically adjusts the task splitting and computational resource allocation in accordance with energy consumption and queue backlogs, and more tasks are offloaded to reduce energy consumption. In Fig. 6(c), the energy consumption performances of EMM and DAC remain unchanged due to the fixed task splitting and resource allocation strategy. With the increase of server-side computing capability, QUARTER can lower energy consumption since

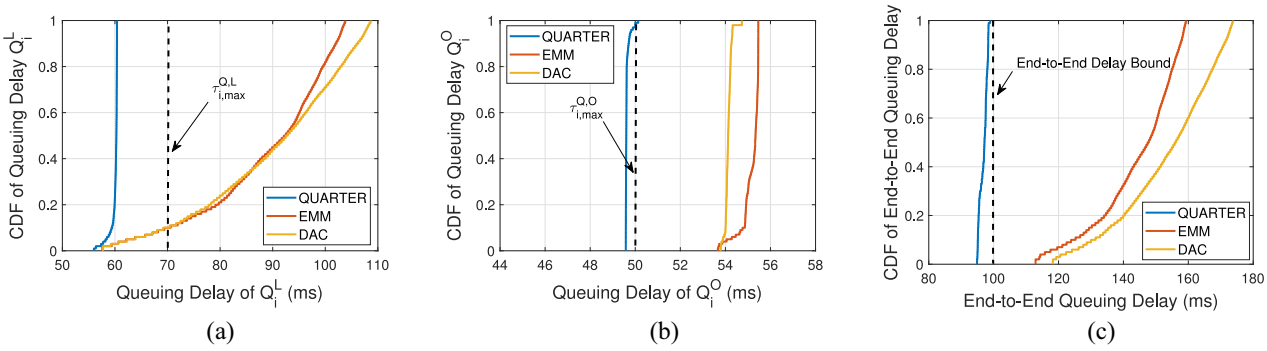

Fig. 7. Performances of queuing delay satisfaction. (a) CDF of queuing delay Q_i^L. (b) CDF of queuing delay Q_i^O. (c) CDF of end-to-end queuing delay.

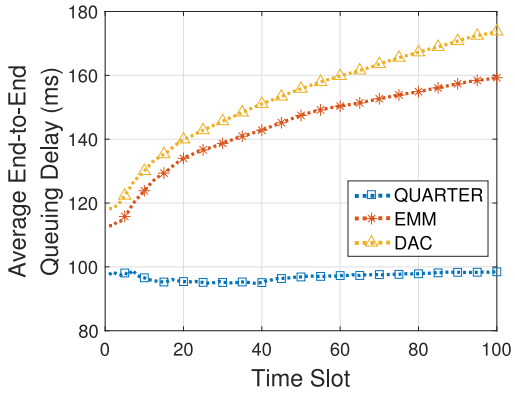

Fig. 8. Average end-to-end queuing delay.

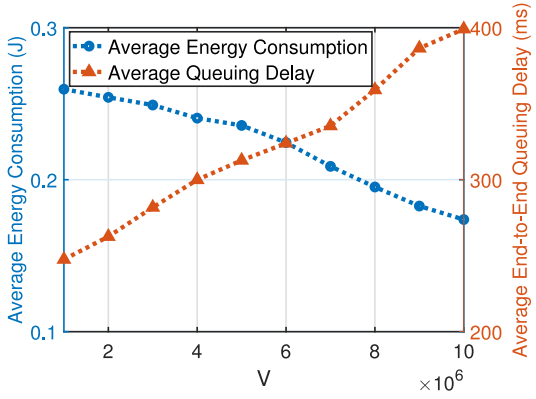

Fig. 9. Impact of V.

more tasks are dynamically split for offloading to make a full utilization of server-side computing capability and thereafter reducing the energy consumption. However, the communication capacity becomes the bottleneck of further energy consumption reduction.

Fig. 7(a)–(c) shows the cumulative distribution function (CDF) plots of the queuing delays of Q_i^L, Q_i^O, and the end-to-end queuing delay, which consists of three parts, i.e., the queuing delays of Q_i^O and $Y_{i,j}$ as well as the queuing delay of result feedback. It can be seen that QUARTER achieves the superior performance in satisfactory probability of queuing delay. Specifically, QUARTER achieves 100%, 99.82%, and 100% satisfaction probability of the queuing delays of Q_i^L, Q_i^O, and the end-to-end queuing delay. DAC performs the worst since the queue unawareness leads to large queue backlogs in the same server and results in larger queuing delay of $Y_{i,j}$. It validates the effectiveness of exploiting queue-aware task offloading to optimize queuing delay performance.

Fig. 8 shows the average end-to-end queuing delay. When $t = 100$, compared with DAC and EMM, QUARTER reduces the end-to-end queuing delay by 43.28% and 38.19%, respectively.

Fig. 9 presents the impact of V. As V increases, QUARTER pays more emphasis to energy consumption minimization rather than queuing delay reduction. By increasing V from 10^6 to 10^7, the average energy consumption is reduced by 33.03% while the average queuing delay is increased by 37.98%. Therefore, QUARTER can dynamically balance the tradeoff between energy consumption minimization and queuing delay reduction.

VII. Conclusion

In this article, we investigated the task offloading and resource allocation problem in SAG-PIoT network. Specifically, QUARTER was proposed to dynamically adjust task offloading and resource allocation in accordance with queue information, and QAC was proposed to cope with the problem of dimensionality curse brought by network heterogeneity and time-varying resource constraints. Simulation results demonstrate that QUARTER can achieve superior performances in energy consumption, queue backlog, and convergence speed. Compared with existing DAC and EMM, the energy consumption of QUARTER is reduced by 22.36% and 23.13%, while the end-to-end queuing delay is reduced by 43.28% and 38.19%, respectively. The simulation results indicate that QUARTER can dynamically tradeoff the energy consumption minimization and queuing delay reduction by adjusting V. In the future, the model robustness can be further improved by exploiting the similar experienced environment observations of nearby PIoT devices to collaboratively train the DRL model.

Appendix
Proof of Theorem 3

For the local-computation queue, based on (2) and (3), we have

$$Q_i^L(t+1) = Q_i^L(t) - U_i^L(t) + A_i^L(t). \tag{53}$$

Likewise, for the device-side task offloading queue and server-side task queue, we have

$$Q_i^O(t+1) = Q_i^O(t) - U_i^O(t) + A_i^O(t) \tag{54}$$

$$Y_{i,j}(t+1) = Y_{i,j}(t) - U_{i,j}^C(t) + x_{i,j}(t)U_i^O(t). \tag{55}$$

Based on (53)–(55), we can derive that

$$\frac{Q_i^L(t+1)}{\tilde{A}_i^L(t+1)} = \frac{Q_i^L(t)}{\tilde{A}_i^L(t+1)} + \frac{A_i^L(t)}{\tilde{A}_i^L(t+1)} - \frac{U_i^L(t)}{\tilde{A}_i^L(t+1)} \tag{56}$$

$$\frac{Q_i^O(t+1)}{\tilde{A}_i^O(t+1)} = \frac{Q_i^O(t)}{\tilde{A}_i^O(t+1)} + \frac{A_i^O(t)}{\tilde{A}_i^O(t+1)} - \frac{U_i^O(t)}{\tilde{A}_i^O(t+1)} \tag{57}$$

$$\frac{Y_{i,j}(t+1)}{\tilde{U}_{i,j}^O(t+1)} = \frac{Y_{i,j}(t)}{\tilde{U}_{i,j}^O(t+1)} + \frac{x_{i,j}(t)U_i^O(t)}{\tilde{U}_{i,j}^O(t+1)} - \frac{U_{i,j}^C(t)}{\tilde{U}_{i,j}^O(t+1)}. \tag{58}$$

Based on the definition of moving time-averaged task arrival rates of local-computation queues, task-offloading queues, and offloaded-task queues, i.e., (17), we can derive that

$$\begin{aligned}\tilde{A}_i^O(t+1) &= \frac{\sum_{m=0}^{t-1} A_i^O(m) + A_i^O(t)}{t+1} \\ &\le \frac{\sum_{m=0}^{t-1} A_i^O(m) + A_i(t)}{t+1}\end{aligned} \tag{59}$$

$$\begin{aligned}\tilde{U}_{i,j}^O(t+1) &= \frac{\sum_{m=0}^{t-1} x_{i,j}(m)U_{i,j}^O(m) + x_{i,j}(t)U_{i,j}^O(t)}{t+1} \\ &\le \frac{\sum_{m=0}^{t-1} x_{i,j}(m)U_{i,j}^O(m) + U_{i,j,\max}^O(t)}{t+1}.\end{aligned} \tag{60}$$

Then, plugging (56)–(60) into (31), the upper bound of the drift-plus-penalty can be converted to

$$\begin{aligned}&\Delta_V L(\Theta(t)) \\ &\le C' + \sum_{m_i\in\mathcal{M}} \mathbb{E}\big[Q_i^L(t)\big(A_i^L(t) - U_i^L(t)\big)|\Theta(t)\big] \\ &\quad + \sum_{m_i\in\mathcal{M}} \mathbb{E}\Big[Q_i^O(t)\Big(A_i^O(t) - U_i^O(t)\Big)|\Theta(t)\Big] \\ &\quad + \sum_{m_i\in\mathcal{M}} \mathbb{E}\Bigg[Z_i^{Q,L}(t)\Bigg(\frac{Q_i^L(t)}{\tilde{A}_i^L(t+1)} + \frac{A_i^L(t)}{\tilde{A}_i^L(t+1)} \\ &\qquad - \frac{U_i^L(t)}{\tilde{A}_i^L(t+1)}\Bigg)|\Theta(t)\Bigg] \\ &\quad + \sum_{m_i\in\mathcal{M}} \mathbb{E}\Bigg[Z_i^{Q,O}(t)\Bigg(\frac{Q_i^O(t)}{\tilde{A}_i^O(t+1)} + \frac{A_i^O(t)}{\tilde{A}_i^O(t+1)} \\ &\qquad - \frac{(t+1)U_i^O(t)}{\sum_{m=0}^{t-1} A_i^O(m) + A_i(t)}\Bigg)|\Theta(t)\Bigg] \\ &\quad + \sum_{m_i\in\mathcal{M}}\sum_{n_j\in\mathcal{N}} \mathbb{E}\Big[Y_{i,j}(t)\Big(x_{i,j}(t)U_i^O(t) - U_{i,j}^C(t)\Big)|\Theta(t)\Big] \\ &\quad + \sum_{m_i\in\mathcal{M}}\sum_{n_j\in\mathcal{N}} \mathbb{E}\Bigg[Z_{i,j}^Y(t)\Bigg(\frac{Y_{i,j}(t)}{\tilde{U}_{i,j}^O(t+1)} + \frac{x_{i,j}(t)U_i^O(t)}{\tilde{U}_{i,j}^O(t+1)} \\ &\qquad - \frac{(t+1)U_{i,j}^C(t)}{\sum_{m=0}^{t-1} x_{i,j}(m)U_{i,j}^O(m) + U_{i,j,\max}^O(t)}\Bigg)|\Theta(t)\Bigg] \\ &\quad + \sum_{m_i\in\mathcal{M}}\sum_{n_j\in\mathcal{N}} \mathbb{E}\Big[Z_{i,j}^{Y,\text{out}}(t)\big(\big[1 - a_{i,j}(t)\big] \\ &\qquad \mathbb{I}\left\{Y_{i,j}(t)f_{i,j}(t) \ne 0\right\}\tau\big)|\Theta(t)\Big] \\ &\quad + V\mathbb{E}[\eta(t)|\Theta(t)]\end{aligned} \tag{61}$$

where $C' = C - \sum_{m_i\in\mathcal{M}} [Z_i^{Q,L}(t)\tau_{i,\max}^{Q,L} + Z_i^{Q,O}(t)\ \tau_{i,\max}^{Q,O} + \sum_{n_j\in\mathcal{N}} Z_{i,j}^Y(t)\ \tau_{i,j,\max}^Y + \sum_{n_j\in\mathcal{N}} Z_{i,j}^{Y,\text{out}}(t)\ \tau_{i,j,\max}^{Y,\text{out}}]$.

This completes the proof of Theorem 3.

REFERENCES

[1] B. Li, Z. Fei, and Y. Zhang, "UAV communications for 5G and beyond: Recent advances and future trends," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 2241–2263, Apr. 2019.

[2] G. Bedi, G. K. Venayagamoorthy, R. Singh, R. R. Brooks, and K. Wang, "Review of Internet of Things (IoT) in electric power and energy systems," *IEEE Internet Things J.*, vol. 5, no. 2, pp. 847–870, Apr. 2018.

[3] M. Tariq, M. Ali, F. Naeem, and H. V. Poor, "Vulnerability assessment of 6G-enabled smart grid cyber-physical systems," *IEEE Internet Things J.*, early access, Dec. 2, 2020, doi: 10.1109/JIOT.2020.3042090.

[4] Y. Liu, M. Peng, G. Shou, Y. Chen, and S. Chen, "Toward edge intelligence: Multiaccess edge computing for 5G and Internet of Things," *IEEE Internet Things J.*, vol. 7, no. 8, pp. 6722–6747, Aug. 2020.

[5] R. Siddavaatam, I. Woungang, G. H. S. Carvalho, and A. Anpalagan, "Mobile cloud storage over 5G: A mechanism design approach," *IEEE Syst. J.*, vol. 13, no. 4, pp. 4060–4071, Dec. 2019.

[6] J. Kwak, Y. Kim, L. B. Le, and S. Chong, "Hybrid content caching in 5G wireless networks: Cloud versus edge caching," *IEEE Trans. Wireless Commun.*, vol. 17, no. 5, pp. 3030–3045, May 2018.

[7] Z. Zhang *et al.*, "6G wireless networks: Vision, requirements, architecture, and key technologies," *IEEE Veh. Technol. Mag.*, vol. 14, no. 3, pp. 28–41, Sep. 2019.

[8] H. Nishiyama, Y. Tada, N. Kato, N. Yoshimura, M. Toyoshima, and N. Kadowaki, "Toward optimized traffic distribution for efficient network capacity utilization in two-layered satellite networks," *IEEE Trans. Veh. Technol.*, vol. 62, no. 3, pp. 1303–1313, Mar. 2013.

[9] F. Zhou, Y. Wu, R. Q. Hu, and Y. Qian, "Computation rate maximization in UAV-enabled wireless-powered mobile-edge computing systems," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 9, pp. 1927–1941, Sep. 2018.

[10] L. Pu, X. Chen, G. Mao, Q. Xie, and J. Xu, "Chimera: An energy-efficient and deadline-aware hybrid edge computing framework for vehicular crowdsensing applications," *IEEE Internet Things J.*, vol. 6, no. 1, pp. 84–99, Feb. 2019.

[11] M. Kamal, G. Srivastava, and M. Tariq, "Blockchain-based lightweight and secured V2V communication in the Internet of Vehicles," *IEEE Trans. Intell. Transp. Syst.*, early access, Jun. 24, 2020, doi: 10.1109/TITS.2020.3002462.

[12] T. Hong, W. Zhao, R. Liu, and M. Kadoch, "Space-air-ground IoT network and related key technologies," *IEEE Wireless Commun.*, vol. 27, no. 2, pp. 96–104, Apr. 2020.

[13] Z. Li *et al.*, "Energy efficient resource allocation for UAV-assisted space-air-ground Internet of Remote Things networks," *IEEE Access*, vol. 7, pp. 145348–145362, Oct. 2019.

[14] B. Shang and L. Liu, "Mobile-edge computing in the sky: Energy optimization for air-ground integrated networks," *IEEE Internet Things J.*, vol. 7, no. 8, pp. 7443–7456, Aug. 2020.

[15] Y. Sun, S. Zhou, and J. Xu, "EMM: Energy-aware mobility management for mobile edge computing in ultra dense networks," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 11, pp. 2637–2646, Nov. 2017.

[16] N. Kiran, C. Pan, S. Wang, and C. Yin, "Joint resource allocation and computation offloading in mobile edge computing for SDN based wireless networks," *J. Commun. Netw.*, vol. 22, no. 1, pp. 1–11, Feb. 2020.

[17] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA, USA: MIT Press, 1998.

[18] Y. Dai, D. Xu, K. Zhang, S. Maharjan, and Y. Zhang, "Deep reinforcement learning and permissioned blockchain for content caching in vehicular edge computing and networks," *IEEE Trans. Veh. Technol.*, vol. 69, no. 4, pp. 4312–4324, Apr. 2020.

[19] N. Cheng *et al.*, "Space/aerial-assisted computing offloading for IoT applications: A learning-based approach," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 5, pp. 1117–1129, May 2019.

[20] S. Zhu, L. Gui, N. Cheng, Q. Zhang, F. Sun, and X. Lang, "UAV-enabled computation migration for complex missions: A reinforcement learning approach," *IET Commun.*, vol. 14, no. 15, pp. 2472–2480, Sep. 2020.
[21] M. J. Neely, *Stochastic Network Optimization With Application to Communication and Queueing Systems*. San Rafael, CA, USA: Morgan and Claypool, 2010.
[22] Q. Hu, Y. Cai, G. Yu, Z. Qin, M. Zhao, and G. Ye Li, "Joint offloading and trajectory design for UAV-enabled mobile edge computing systems," *IEEE Internet Things J.*, vol. 6, no. 2, pp. 1879–1892, Apr. 2019.
[23] Z. Zhou, H. Liao, X. Zhao, B. Ai, and M. Guizani, "Reliable task offloading for vehicular fog computing under information asymmetry and information uncertainty," *IEEE Trans. Veh. Technol.*, vol. 68, no. 9, pp. 8322–8335, Sep. 2019.
[24] Z. Zhang, F. Yu, F. Fu, Q. Yan, and Z. Wang, "Joint offloading and resource allocation in mobile edge computing systems: An actor-critic approach," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Abu Dhabi, UAE, Dec. 2019, pp. 1–6.
[25] C.-F. Liu, M. Bennis, M. Debbah, and H. V. Poor, "Dynamic task offloading and resource allocation for ultra-reliable low-latency edge computing," *IEEE Trans. Commun.*, vol. 67, no. 6, pp. 4132–4150, Jun. 2019.
[26] A. Al-Hourani, S. Kandeepan, and S. Lardner, "Optimal LAP altitude for maximum coverage," *IEEE Wireless Commun. Lett.*, vol. 3, no. 6, pp. 569–572, Dec. 2014.
[27] Y. Mao, J. Zhang, S. H. Song, and K. B. Letaief, "Power-delay tradeoff in multi-user mobile-edge computing systems," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Washington DC, USA, Sep. 2016, pp. 1–6.
[28] J. D. C. Little, "A proof for the queuing formula: L = λW," *Oper. Res.*, vol. 9, no. 3, pp. 383–387, Jun. 1961.
[29] Z. Zhou, Y. Guo, Y. He, X. Zhao, and W. M. Bazzi, "Access control and resource allocation for M2M communications in industrial automation," *IEEE Trans. Ind. Informat.*, vol. 15, no. 5, pp. 3093–3103, May 2019.
[30] W. Bao, H. Chen, Y. Li, and B. Vucetic, "Joint rate control and power allocation for non-orthogonal multiple access systems," *IEEE J. Sel. Areas Commun.*, vol. 35, no. 12, pp. 2798–2811, Dec. 2017.
[31] Z. Zhou, K. Ota, M. Dong, and C. Xu, "Energy-efficient matching for resource allocation in D2D enabled cellular networks," *IEEE Trans. Veh. Technol.*, vol. 66, no. 6, pp. 5256–5268, Jun. 2017.
[32] S. Y. Jung, J. Hong, and K. Nam, "Current minimizing torque control of the IPMSM using Ferrari's method," *IEEE Trans. Power Electron.*, vol. 28, no. 12, pp. 5603–5617, Dec. 2013.
[33] K. T. K. Cheung, S. Yang, and L. Hanzo, "Achieving maximum energy-efficiency in multi-relay OFDMA cellular networks: A fractional programming approach," *IEEE Trans. Commun.*, vol. 61, no. 7, pp. 2746–2757, Jul. 2013.
[34] C. Qiu, H. Yao, F. R. Yu, F. Xu, and C. Zhao, "Deep *Q*-learning aided networking, caching, and computing resources allocation in software-defined satellite-terrestrial networks," *IEEE Trans. Veh. Technol.*, vol. 68, no. 6, pp. 5871–5883, Jun. 2019.
[35] K. S. Yildirim, "Gradient descent algorithm inspired adaptive time synchronization in wireless sensor networks," *IEEE Sensors J.*, vol. 16, no. 13, pp. 5463–5470, Jul. 2016.
[36] R. S. Sutton, D. Mcallester, S. Singh, and Y. Mansour, "Policy gradient methods for reinforcement learning with function approximation," in *Proc. 12th Int. Conf. Neural Inf. Process. Syst.*, Denver, CO, USA, Nov. 1999, pp. 1057–1063.

Haijun Liao (Graduate Student Member, IEEE) is pursuing the Ph.D degree with the School of Electrical and Electronic Engineering, North China Electric Power University, China.

Her research interests mainly focus on resource allocation in smart grid communications and Internet of Things (IoT).

She was the recipient of the IEEE International Conference on Wireless Communications 2019 Best Paper Award and the IEEE VTC-2020 Spring Best Student Paper Award.

Zhenyu Zhou (Senior Member, IEEE) received the M.E. and Ph.D degrees from Waseda University, Tokyo, Japan, in 2008 and 2011, respectively.

From 2012 to 2019, he was an Associate Professor with the School of Electrical and Electronic Engineering, North China Electric Power University, China. Since 2019, he has been a full professor at the same university. His research interests mainly focus on resource allocation in device-to-device (D2D) communications, machine-to-machine (M2M) communications, smart grid communications, and Internet of Things (IoT).

Dr. Zhou served as an Associate Editor for IEEE INTERNET OF THINGS JOURNAL, IEEE ACCESS, *EURASIP Journal on Wireless Communications and Networking*, and a Guest Editor for IEEE COMMUNICATIONS MAGAZINE, IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS AND TRANSACTIONS ON EMERGING TELECOMMUNICATIONS TECHNOLOGIES. He was the recipient of the IET Premium Award in 2017, the IEEE GLOBECOM 2018 Best Paper Award, the IEEE INTERNATIONAL WIRELESS COMMUNICATIONS AND MOBILE COMPUTING CONFERENCE (IWCMC) 2019 Best Paper Award, and the IEEE COMMUNICATIONS SOCIETY ASIA-PACIFIC BOARD OUTSTANDING YOUNG RESEARCHER. He is a senior member of the Chinese Institute of Electronics, and China Institute of Communications.

Xiongwen Zhao (Senior Member, IEEE) received the Ph.D. degree (Hons.) from the Helsinki University of Technology (TKK), Finland, in 2002.

From 1992 to 1998, he was with the Laboratory of Communications System Engineering, China Research Institute of Radiowave Propagation, where he was the Director and Senior Engineer. From 1999 to 2004, he was with the Radio Laboratory, TKK, as a Senior Researcher and a Project Manager in the areas of MIMO channel modeling and measurements at 2, 5, and 60 GHz as well as UWB. From 2004 to 2011, he was with Elektrobit Corporation, Espoo, Finland, as a Senior Specialist, EB Wireless Solutions. From 2004 to 2007, he worked in the European WINNER Project as a Senior Researcher in MIMO channel modeling for 4G radio systems. From 2006 to 2008, he also worked in the field of wireless network technologies such as WiMAX and wireless mesh networks. From 2008 to 2009, he worked in mobile satellite communications for GMR-1 3G, DVB-SH RF link budget, and antenna performance evaluations. He is currently a responsibility Professor in Information and Communication Discipline at North China Electric Power University, Beijing, and chairs several projects at the National Science Foundation of China, the Key Program of the Beijing Municipal Natural Science Foundation, Beijing Municipal Science and Technology Commission, and the State Key Laboratories and Industries on radio channel and wireless power communications research. His research interests mainly focus on 5G mobile communication, MIMO communication, wireless channels, Internet of Things (IoT), antenna arrays, microcellular radio, microstrip antenna arrays, millimeter wave propagation.

He is a Fellow of the Chinese Institute of Electronics. He was a recipient of the IEEE Vehicular Technology Society Neal Shepherd Memorial Best Propagation Paper Award, in 2014. He served as a TPC Member, the Session Chair, and a Keynote Speaker for numerous international and national conferences.

Yang Wang is currently the Secretary General of the Energy and Internet Committee, China Communications Society, and the Chairman of the Information and Communication Committee, China Electricity Union. He is the Deputy Director of the Institute of Information and Communication, China Electric Power Research Institute Co. Ltd, Director of 5G Center, Research Institute of Energy and Internet Technology, State Grid Corporation of China, and the professorate senior engineer. He has been engaged in scientific research in the field of electric power information and communication technology for a long time. He has carried out in-depth research in 5G wireless network communication technology, electric power Beidou application technology, and testing and evaluation for electric power information and communication system. He is the leader of the electric power information and communication professional scientific research team, State Grid Corporation of China.

He has been responsible to implement many national and corporate level major science and technology project, participated in the construction of "next generation Internet", "Internet of Things application" and other national level demonstration projects, and led the establishment of the only authoritative inspection agencies about the information communication hardware products into the online network and information system of State Grid Corporation of China, realizing a complete coverage of electric power information communication professional certification testing.