Green Parallel Online Offloading for DSCI-Type Tasks in IoT-Edge Systems

Junqi Chen, Huaming Wu, *Senior Member, IEEE*, Ruidong Li, *Senior Member, IEEE*, and Pengfei Jiao, *Member, IEEE*

***Abstract*—In order to meet people's demands for intelligent and user-friendly Internet of Things (IoT) services, the amount of computation is increasing rapidly and the requirements of task delay are becoming increasingly more stringent. However, the constrained battery capacity of IoT devices greatly limits the user experience. Energy harvesting technologies enable green energy to provide continuous energy support for devices in the IoT environment. Together with the maturity of the mobile edge computing technology and the development of parallel computing, it provides a strong guarantee for the normal operation of resource-constrained IoT devices. In this article, we design a parallel offloading strategy based on Lyapunov optimization, which is conducive to efficiently finding the optimal decision for delay-sensitive and compute-intensive tasks. We establish a stochastic optimization problem on a discrete-time slot system and propose a green parallel online offloading algorithm (GPOOA). By decoupling the target problem three times, the joint optimization of green energy, task division factor, CPU frequency, and transmission power is realized. Experimental results demonstrate that under the constraints of strict task deadlines and limited server computing resources, GPOOA performs well in terms of system cost and task drop ratio, far superior to several existing offloading algorithms.**

***Index Terms*—Energy harvesting (EH), Internet of Things (IoT), mobile edge computing (MEC), perturbed Lyapunov optimization, task offloading.**

Nomenclature

$A(L,\tau)$	Unit task.
τ_0	Slot length of the system.
τ'	Temporary time variable.
$I_{i,l}^t, I_{i,e}^t, I_{i,d}^t$	Task division factors.
$T_{i,l}^t$	Delay to process a unit task locally.
$T_{i,e}^t$	Delay to offload a unit task.
$E_{i,l}^t$	Energy consumption to process a unit task locally.
$E_{i,e}^t$	Energy consumption to offload a unit task.
$f_{\text{local}}^{\max}$	Maximum CPU-cycle frequency of local devices.
K	Number of CPU cycles required for a unit task.
$E_H^{\max}$	Maximum energy can grab from the outside.
$E^{\max}$	Maximum discharge energy of the battery.
$p^{\max}$	Maximum transmission power allowed by the device.
b_i^t	Battery level of the ith device in the time slot t.
χ_i^t	Task drop indicator.
ζ_i^t	Task generation indicator.
ψ	Penalty weight (the system cost of dropping the task).
D_i^t	Delay of the ith device in the time slot t.
e_i^t	Green energy level collected through the energy harvester in the time slot t.
ε_i^t	Total energy consumption of the ith device in the time slot t.
Q	Maximum number of devices that the edge server can connect to in a time slot.

Manuscript received 13 December 2021; revised 20 March 2022; accepted 5 April 2022. Date of publication 19 April 2022; date of current version 9 September 2022. This work was supported in part by the National Natural Science Foundation of China under Grant 62071327 and Grant 61902278 and in part by JSPS KAKENHI under Grant 19H04105. Paper no. TII-21-5530. *(Corresponding author: Huaming Wu.)*

Junqi Chen and Huaming Wu are with the Center for Applied Mathematics, Tianjin University, Tianjin 300072, China (e-mail: junqichen@tju.edu.cn; whming@tju.edu.cn).

Ruidong Li is with the Institute of Science and Engineering, Kanazawa University, Kanazawa 920-1192, Japan (e-mail: liruidong@ieee.org).

Pengfei Jiao is with the School of Cyberspace, Hangzhou Dianzi University, Hangzhou 310018, China (e-mail: pjiao@tju.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TII.2022.3167668.

Digital Object Identifier 10.1109/TII.2022.3167668

I. Introduction

FIFTH-GENERATION (5G) mobile communication has paved the way for the rapid proliferation of the Internet of Things (IoT) [1]. With the increasingly diversified and user-friendly functions of IoT devices, various compute-intensive (CI) and delay-sensitive (DS) applications have emerged, e.g., augmented reality, virtual reality, speech recognition, video analysis [2], and smart homes. The underlying IoT tasks generated by these applications usually require high computational demands and short delays [3], which are referred to as DS and CI (DSCI) tasks [4]. In most cases, the limited computing resources and battery capacity of the device itself are difficult to support DSCI-type tasks. This can easily lead to tasks not being executed smoothly due to battery depletion or long response times. Regardless of transmission latency, it is ideal to offload the workload to a cloud server with abundant computing resources for processing. However, it is unrealistic to offload all tasks to remote clouds. On the one hand, large-scale and long-distance transmission of tasks will consume a lot of energy. On the other hand, frequent communication with the cloud may also cause greater communication delays [5]. As a result, not only has today's already congested network become worse [6], [7], but the entire IoT system has also become unstable.

The emergence of mobile edge computing (MEC) has made up for the deficiencies of cloud computing and can support the needs of mission-critical computing for low latency, intensive computing, and mass storage [8], [9]. However, there exist several bottlenecks restricting the further development of the IoT technology. For instance, battery life has become one of the main factors affecting user experience. Limited battery life increases the maintenance cost of IoT devices, and the cost of replacing batteries is often higher than the cost of IoT devices themselves. For instance, in an industrial environment with only 10 000 sensors, the battery needs to be replaced nearly 3333 times each year [10]. Not to mention how to deal with today's huge IoT system where everything is interconnected. Fortunately, energy harvesting (EH), a promising technology that obtains harvested green energy from the external environment (e.g., solar and wind energy) and converts the captured renewable energy into electrical energy through an EH device, provides a new opportunity for powering IoT-edge systems [11]. The working range of most IoT devices and sensors is between 0.1 μW and 1 W, which can be easily handled by EH devices [12]. While EH extends the life cycle of the equipment, it also eliminates the limitation of fixed rechargeable batteries as energy sources. Despite the obvious advantages of using green energy for power supply, the energy collection process is highly intermittent and random, which poses a huge challenge for making full use of green energy. In addition, although the edge server has more abundant computing resources than the device itself, e.g., faster CPU frequency and higher parallel computing power [13]. However, most MEC servers in the real world have limited computing capacities and cannot match cloud computing, especially in a multidevice IoT environment.

To address the abovementioned challenges, several Lyapunov optimization-based solutions, e.g., DBWA [14] and EEDTO [15], have been proposed to minimize system energy consumption by optimizing the workload distribution based on the IoT-edge-cloud computing architecture. Chen *et al.* [16] transformed the energy minimization problem into a knapsack problem and proposed an energy-efficient dynamic offloading algorithm (DOA), which can approximate the minimum transmission energy consumption while ensuring the stability of the system. Although the aforementioned approaches strive for energy-efficient algorithms, they do not take into account green EH techniques or the mobility of the devices. Using execution delays and task failures as execution costs, Mao *et al.* [17] developed a low-complexity Lyapunov optimization-based dynamic computation offloading (LODCO) algorithm for models from a single device to a single edge. Zhao *et al.* [13] inherited the advantages of the LODCO algorithm and migrated it to the multidevice multiserver scenario, which is more in line with the real world. Inspired by the abovementioned practices, we try to apply the Lyapunov optimization technique combined with the mobility of the device in the multidevice multiserver model.

Regarding how to reduce latency to meet the needs of different types of tasks, Yousefpour *et al.* [18] utilized the concept of load sharing to reduce service latency by sharing load among fog nodes. Mukherjee *et al.* [19] used quadratic constrained quadratic programming to solve the DS task offloading problem when considering local execution delay and transmission delay. Liu *et al.* [20] developed an efficient 1-D search algorithm to solve the power-constrained delay minimization problem under different time scales. Instead, in this article, we use parallel offloading for DSCI-type tasks to achieve this goal.

Green computing and communication have become the new darlings of researchers. Taking advantage of EH and device-to-device communication, Zhou [21] proposed GreenEdge, a novel framework for sustainable edge computing, and verified its feasibility. Deng *et al.* [22] designed a green sustainable MEC framework for dynamic and parallel computing offloading and energy management algorithms. However, this work was carried out under the ideal state of MEC server computing resources, ignoring the mobility of IoT devices. We consider more scenarios where the edge server has limited computing resources and the devices can move freely. Hu *et al.* [23] proposed an online mobile-aware offloading and resource allocation algorithm, which combined Lyapunov optimization and semidefinite programming methods. Although the task drop rate and migration cost are considered, the main optimization is the total energy consumption. Inspired by the above, we attempt to apply the EH technology to deal with system energy consumption, while placing more emphasis on the optimization of latency.

In this article, we design a green parallel online offloading algorithm (GPOOA) for DSCI-type tasks. GPOOA is based on the Lyapunov optimization framework to offload tasks in a parallel manner in multidevice and multiserver scenarios, and combines the EH technology to power the devices. Our goal is to optimize the user experience and system robustness by reducing system costs. The main contributions of this article are summarized as follows.

1) We establish a multidevice and multiserver MEC system model for DSCI-type tasks, and formally define the stochastic optimization problem on a discrete-time slot system, taking the mobility of the device into account.
2) To meet the delay requirements of DSCI-type tasks, we propose a parallel offloading strategy with the close collaboration between IoT devices and edge servers. To ensure the robustness of task processing, we take the drop ratio into consideration to motivate the system to perform tasks as many as possible.
3) We apply the EH technology to IoT devices to make full use of the advantages of green energy, and further propose a Lyapunov-guided solution to maintain the continuity of energy supply in IoT-edge systems. Meanwhile, we decoupled the optimization problem three times and designed the GPOOA algorithm.

II. System Model and Problem Formulation

In this section, we first introduce the system model, computation offloading, and EH models, and then formulate

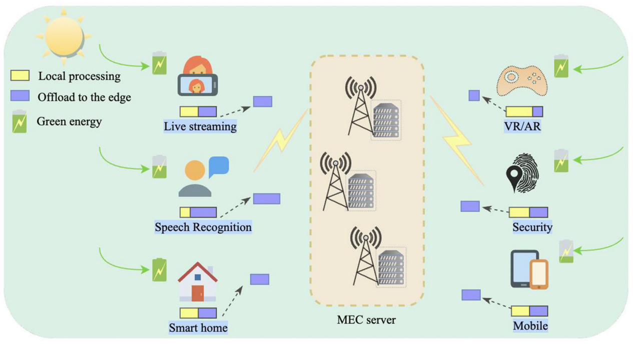

Fig. 1. System model with the EH technology.

the online decision problem for task offloading in MEC environments.

A. System Model

As shown in Fig. 1, we consider a 5G-based MEC environment with multiple IoT devices and multiple MEC servers, where IoT devices $\mathcal{M} = \{1, 2, \ldots, M\}$ can move freely in the environment, while MEC servers $\mathcal{N} = \{1, 2, \ldots, N\}$ are statically deployed. According to the EH technology, each device in the scenario is equipped with an energy harvester. The energy harvester will collect green energy (e.g., solar and wind) from the environment and convert it into electricity to power the device itself. According to the divisible load theory [24], we assume that the unit task $A(L, \tau)$ generated by the device is divisible and of DSCI type, where L (in bit) is the task size and τ (in ms) is the deadline of the task. Divisible here means that the task $A(L, \tau)$ can be divided into two parts arbitrarily. And, we adopt the cooperation of local devices and edge servers to process tasks in parallel.

We perform parallel offloading on a discrete-time slot system, where the time slot set is $\mathcal{T} = \{0, 1, \ldots, T-1\}$ with the slot length τ_0. Tasks are randomly generated with the Bernoulli distribution in the time slots, and the task arrival rate is defined as ρ $(0 \le \rho \le 1)$. Let ζ_i^t denote a task generation indicator and $\zeta_i^t = 1$ means that the ith IoT device has a task generated in the time slot t. And, $\zeta_i^t = 0$ indicates that no task is generated. In addition, we define the task division factors of ith device in time slot t as follows:

$$I_{i,l}^t + I_{i,e}^t + I_{i,d}^t = 1 \quad (1)$$

$$I_{i,l}^t, I_{i,e}^t \in [0, 1], \; I_{i,d}^t \in \{0, 1\} \quad (2)$$

where $I_{i,l}^t$ and $I_{i,e}^t$ indicate the ratio of the task processed locally and offloaded to the edge server, respectively. The value of $I_{i,d}^t$ is either 1 or 0, indicating that the task is either completely discarded or executed. The symbols and their definitions commonly used in this article are summarized in nomenclature.

B. Computation Offloading and EH Models

1) Local Execution Model: The dynamic voltage and frequency scaling (DVFS) technology [25] can adjust execution time and energy consumption by controlling the CPU cycle frequency to achieve low power consumption. Using DVFS, the local execution delay of the ith device in time slot t can be obtained by

$$T_{i,l}^t = \sum_{k=1}^{K} \left(f_{i,k}^t\right)^{-1} \quad \forall t \in \mathcal{T} \quad \forall i \in \mathcal{M} \quad (3)$$

where $K = LW$ is the number of CPU cycles required for a unit task $A(L, \tau)$, W is the number of CPU cycles required to perform one bit locally, and $f_{i,k}^t$ is the frequency allocated by the ith device to the kth CPU cycle in the time slot t. The corresponding local execution energy consumption is

$$E_{i,l}^t = \theta \sum_{k=1}^{K} \left(f_{i,k}^t\right)^2 \quad \forall t \in \mathcal{T}, \quad \forall i \in \mathcal{M} \quad (4)$$

where θ is the capacitance constant [17] that depends on the chip architecture. Here, $f_{i,k}^t \le f_{\text{local}}^{\max} \;\; \forall k \in \{1, 2, \ldots, K\}$, where $f_{\text{local}}^{\max}$ (in cycle/s) represents the maximum CPU-cycle frequency of local devices.

2) MEC Offloading Model: The Shannon–Hartley formula shows that the channel transmission rate is determined by the channel gain $h_{i,j}^t$ and the transmission power p_i^t of the device. So, the transmission rate $v_{i,j}^t$ from the ith device to the jth MEC server can be expressed as [26]

$$v_{i,j}^t = v\left(h_{i,j}^t, p_i^t\right) = \omega \log_2 \left(1 + \frac{h_{i,j}^t p_i^t}{\sigma}\right) \quad (5)$$

where $h_{i,j}^t = \gamma_{i,j}^t g_0 (\frac{d_0}{d_{i,j}^t})^\alpha$ represents the channel gain from the ith device to the jth MEC server, $\gamma_{i,j}^t$ is the small-scale fading channel power gain, d_0 represents the reference distance, $d_{i,j}^t$ is the distance from the device i to the MEC server j, g_0 is the pass-loss constant, α is the pass-loss exponent, the bandwidth of the channel is denoted as ω, and σ is the noise power at the MEC server.

Typically, the downlink transmission rate is much higher than the uplink rate. And, the size of the output result is usually much smaller than the input, so our model ignores the return time of the result. Inspired by [13], [17], and [22], our model inherits the delay assumptions in these works. That is, we do not consider the execution delay of the MEC server for simplicity. If a unit task generated by the ith device is processed by the jth MEC server, the corresponding offloading delay is calculated by

$$T_{i,e}^t = \frac{L}{v_{i,j}^t} \quad \forall t \in \mathcal{T}, i \in \mathcal{M}. \quad (6)$$

And, the corresponding energy consumed by the ith device of offloading tasks is

$$E_{i,e}^t = p_i^t T_{i,e}^t \quad \forall t \in \mathcal{T}, \quad \forall i \in \mathcal{M}. \quad (7)$$

3) EH Model: We adopt the EH technology to make full use of green energy to provide energy support for IoT devices. The energy harvester converts green energy, such as solar and wind, and mechanical energy obtained from the outside into electrical energy and stores it in the battery to ensure the normal operation of the device. However, the process of obtaining green energy in the real world is stochastic.

Assuming that the green energy arrives at the ith device in the time slot t with $E_{i,H}^t$, which is independent and identically distributed, and $E_{i,H}^t \leq E_H^{\max}$, were, $E_H^{\max}$ is the maximum energy that the device can grab from the outside world. Define the green energy level collected through the energy harvester of the ith device in the time slot t as e_i^t, and the captured energy e_i^t cannot exceed the randomly arrived green energy level:

$$0 \leq e_i^t \leq E_{i,H}^t. \tag{8}$$

In our model, the generated tasks are either executed in parallel or dropped [satisfy (1)]. Dropping tasks will not generate energy consumption. Therefore, the total energy consumption of the ith device in the time slot t, consists of two parts: a) local part $I_{i,l}^t E_{i,l}^t$ and b) edge part $I_{i,e}^t E_{i,e}^t$, is

$$\varepsilon_i^t = I_{i,l}^t E_{i,l}^t + I_{i,e}^t E_{i,e}^t. \tag{9}$$

In order to prolong the service life of the battery and prevent the battery from overdischarging, the battery output energy of each time slot should not exceed $E^{\max}$

$$0 \leq \varepsilon_i^t \leq E^{\max} \tag{10}$$

where $E^{\max}$ is the maximum discharge energy of the battery in each time slot.

Define the battery level of the ith device in the time slot t as b_i^t. It needs to be emphasized that the energy consumed in each time slot cannot exceed the current battery level, which satisfies

$$\varepsilon_i^t \leq b_i^t < \infty. \tag{11}$$

In other words, if the energy consumed by the processing task exceeds the current battery level, the system will drop the task due to insufficient energy supply. In summary, the battery level of device i in time slot $t+1$ is updated according to

$$b_i^{t+1} = b_i^t - \varepsilon_i^t + e_i^t. \tag{12}$$

C. Problem Formulation

Our goal is to optimize user experience (reduce delay) and improve system stability (reduce task drop ratio) by reducing the total system cost. For DSCI-type tasks, we use a parallel offload strategy to cope with DS features, and we use EH techniques to power devices for CI features. Before defining the system cost, we first define a task drop indicator as: $\chi_i^t = \zeta_i^t I_{i,d}^t$ (if the ith device has a task to generate in time slot t while it is dropped).

Based on the characteristics of the parallel computing framework, the delay of the ith device in time slot t is larger than that of the delays between the local side $I_{i,l}^t T_{i,l}^t$ and the offload to the edge server side $I_{i,e}^t T_{i,e}^t$

$$D_i^t = \zeta_i^t \cdot \max\left\{I_{i,l}^t T_{i,l}^t, I_{i,e}^t T_{i,e}^t\right\}. \tag{13}$$

The cost of the ith device in time slot t is defined as the weighted sum of the delay D_i^t and the task drop indicator χ_i^t

$$\text{cost}_i^t = D_i^t + \psi \chi_i^t \tag{14}$$

where ψ (in s) is the penalty weight, i.e., the system cost of dropping the task. Therefore, the total cost of the system in time slot t is

$$\text{cost}_{\text{total}}^t = \sum_{i=1}^{M} \text{cost}_i^t. \tag{15}$$

Considering that there are many random factors in the system, such as the arrival of tasks, the location of devices, the state of the channel, and the EH situation. First, we formulate the problem as a random optimization problem. And, we want to minimize the response time and task drop ratio in the sense of time average through resource allocation and parallel offloading. So, we first get optimization problem $\mathcal{P}_1$

$$\boldsymbol{\mathcal{P}_1}: \quad \min \lim_{T\to\infty} \frac{1}{T}\mathbb{E}\left[\sum_{t=0}^{T-1} \text{cost}_{total}^t\right]$$

$$\text{s.t.}: \quad (1),\ (2),\ (8),\ (10),\ \text{and}(11)$$

$$D_i^t \leq \tau \tag{16}$$

$$I_{i,l}^t + I_{i,e}^t \leq \zeta_i^t \tag{17}$$

$$0 \leq f_{i,k}^t \leq f_{\text{local}}^{\max} \quad \forall t \in \mathcal{T} \quad \forall i \in \mathcal{M} \tag{18}$$

$$0 \leq p_i^t \leq p^{\max} \quad \forall t \in \mathcal{T} \quad \forall i \in \mathcal{M} \tag{19}$$

where (1) and (2) are the task division factor constraints. Equations (8), (10), and (11) are the energy consumption constraints (the task will be dropped if the energy consumed to perform the task exceeds the current battery level). Equation (16) indicates that the response time constraints (the task will be dropped if it is not executed before the deadline). Equation (17) indicates that the parallel offloading can only occur when there are tasks generated. Equations (18) and (19) are the constraints of the device's CPU frequency and transmission power, respectively, where $p^{\max}$ represents the maximum transmission power allowed by the device.

D. Modified System Cost Minimization Problem

It is worth noting that the energy constraint in (11) makes the system coupled between different time slots when making decisions, which is challenging to directly apply the traditional Lyapunov method. In order to eliminate this coupling effect, similar to [17], we introduce a nonzero energy consumption $E^{\min}$ as the minimum discharge energy of the battery in each time slot to tighten the constraints of the problem $\mathcal{P}_1$, so that an improved stochastic optimization problem can be obtained by

$$\boldsymbol{\mathcal{P}_2}: \quad \min \lim_{T\to\infty} \frac{1}{T}\mathbb{E}\left[\sum_{t=0}^{T-1} \text{cost}_{\text{total}}^t\right]$$

$$\text{s.t.}: \quad (1),\ (2),\ (8),\ (11),\ \text{and}\ (16)\text{–}(19)$$

$$\varepsilon_i^t \in \{0\} \cup \left[E^{\min}, E^{\max}\right] \tag{20}$$

where the constraints of $\mathcal{P}_2$ are much stricter than that of $\mathcal{P}_1$. By forcing $E^{\min}$ to tend to zero, the optimal solution of $\mathcal{P}_2$ will tend to that of $\mathcal{P}_1$. The relationship between the optimal solutions of the two problems is as shown in Lemma 1. According to Lemma 1, we transform how to solve problem $\mathcal{P}_1$ into how to solve problem $\mathcal{P}_2$.

Lemma 1: Let the optimal value corresponding to the optimal solution of the problem $\mathcal{P}_1$ and the problem $\mathcal{P}_2$ be $\mathcal{SP}_1^*$ and $\mathcal{SP}_2^*$, respectively, then we have $\mathcal{SP}_1^* \leq \mathcal{SP}_2^* \leq \mathcal{SP}_1^* + \sum_{m=1}^{M}(\psi - \tau_m^{\min}) \cdot 1_{\{E^{\min} > E_{\tau,\mathrm{m}}^{\min}\}}$, where $\tau_m^{\min} = \min_\tau\{\arg\{E_{\tau,l,m}^{\min} = E^{\min}\}, \arg\{E_{\tau,e,m}^{\min} = E^{\min}\}\}$, $E_{\tau,e,m}^{\min} = \tau\sigma\frac{2^{\frac{LI_{m,e}^t}{\omega\tau}}-1}{h_{i,j}^t}$, and $E_{\tau,l,m}^{\min} = \frac{\theta K^3 I_{m,l}^t}{\tau^2}$.

Proof: Since the constraint condition of $\mathcal{P}_2$ is more stringent than that of $\mathcal{P}_1$, it is easy to draw $\mathcal{SP}_1^* \leq \mathcal{SP}_2^*$. The other side of the inequality can be obtained by constructing a feasible solution of $\mathcal{P}_2$ according to the problem $\mathcal{P}_1$. Let $\langle Q_m^{P_1}(t)\rangle$ be the optimal solution of the mth IoT device in the value space $\mathcal{SP}_1$, the following is to construct an optimal solution in the value space $\mathcal{SP}_2$ for each IoT device. We have, when $\varepsilon(Q_m^{P_1}(t)) \in \{0\} \cup [E^{\min}, E^{\max}]$, let $\langle Q_m^{P_1}(t)\rangle = \langle Q_m^{P_2}(t)\rangle$. Next, we will discuss the situation when $\varepsilon(Q_m^{P_1}(t)) \in (0, E^{\min})$. For the optimization problem $\mathcal{P}_2$, the energy consumption at this time is not within its constraint range, and the tasks in this state will be dropped, i.e., the corresponding optimization value $\mathcal{SP}_2 = \psi$.

Let the minimum energy consumption of the mth IoT device meet the task deadline τ as $E_{\tau,m}^{\min} = E_{\tau,l,m}^{\min} + E_{\tau,e,m}^{\min}$, where $E_{\tau,e,m}^{\min} = \tau\sigma\frac{2^{\frac{LI_{m,e}^t}{\omega\tau}}-1}{h_{i,j}^t}$ represents the minimum energy consumption required for the task to be executed on the device side and $E_{\tau,l,m}^{\min} = \frac{\theta K^3 I_{m,l}^t}{\tau^2}$ represents the minimum energy consumption required for the task to be executed on the MEC side. If $E^{\min} \geq E_{\tau,m}^{\min}$, the newly constructed task drop cost is set to ψ and the corresponding cost of $\mathcal{P}_1$ is at least $\tau_m^{\min}$, where $\tau_m^{\min} = \min_\tau\{\arg\{E_{\tau,l,m}^{\min} = E^{\min}\}, \arg\{E_{\tau,e,m}^{\min} = E^{\min}\}\}$. Thus, the optimal values of these two problems may differ by $\psi - \tau_m^{\min}$ at most. If $E^{\min} < E_{\tau,m}^{\min}$, the generated task will also be dropped for $\mathcal{P}_1$. In summary, there is no difference between the optimal values of the two problems in a discrete-time slot system.

III. Perturbed Lyapunov Optimization-Based Approach

A. Lyapunov Optimization Framework

Because Lyapunov optimization does not require many *a priori* parameters, it can realize real-time control in dynamic systems with relatively low algorithm complexity, which is in line with the characteristics of task generation and the characteristics of capturing green energy. In our model, Lyapunov optimization is combined with parallel offloading and EH, this method does not directly calculate the optimal value, but uses an upper bound to guarantee the stability of the system.

The energy flow is constructed as an energy queue to provide continuous and stable energy support for the normal operation of devices. Each energy queue corresponds to a virtual queue, defined as

$$\tilde{b}_i^t = b_i^t - \beta \quad \forall t \in \mathcal{T} \quad \forall i \in \mathcal{M} \tag{21}$$

where the virtual queue vector formed by all IoT devices is $\tilde{B}^t \triangleq [\tilde{b}_1^t, \tilde{b}_2^t, \ldots, \tilde{b}_M^t]$. The disturbance parameter β of IoT devices with the EH technology is a bounded constant that satisfies

$$\beta \geq \tilde{E}^{\max} + \frac{V\psi}{E^{\min}} \tag{22}$$

where $\tilde{E}^{\max} \triangleq \min\{\max_i\{\varepsilon_i^t\}, E^{\max}\} = \min\{\max\{\beta K (f_{\text{local}}^{\max})^2, p^{\max}\tau_0\}, E^{\max}\}$ is the upper bound of the available energy and V is the nonnegative weight control parameter.

Then, define the Lyapunov function of the virtual energy queue as

$$\mathcal{L}(t) = \frac{1}{2}\sum_{i=1}^{M}\left(\tilde{b}_i^t\right)^2 = \frac{1}{2}\sum_{i=1}^{M}\left(b_i^t - \beta\right)^2 \quad \forall t \in \mathcal{T}. \tag{23}$$

Next, we introduce a one-step conditional Lyapunov drift function to push the quadratic Lyapunov function to a bounded level to form a stable virtual queue, which is formulated as

$$\Delta(t) = \mathbb{E}\left[\mathcal{L}(t+1) - \mathcal{L}(t) \mid \tilde{B}^t\right] \quad \forall t \in \mathcal{T}. \tag{24}$$

Finally, by combining the queue stability with the system cost required to execute the task, we obtain a Lyapunov drift plus penalty function

$$\Delta_V(t) = \Delta(t) + V\mathbb{E}\left[\text{cost}_{\text{total}}^t \mid \tilde{B}^t\right] \quad \forall t \in \mathcal{T}. \tag{25}$$

The parameter V here is consistent with (22), which shows the tradeoff relationship between the energy queue backlog and the system cost. Use the classic Lyapunov technique [27] to scale the upper bound of (25), we have

$$\begin{aligned}\Delta_V(t) \leq\; & \mathbb{E}\left[\sum_{i=1}^{M}\left(\tilde{b}_i^t\left(e_i^t - \varepsilon_i^t\right)\right) \mid \tilde{B}^t\right] + C \\ & + V\mathbb{E}\left[\sum_{i=1}^{M}\left(D_i^t + \psi\chi_i^t\right) \mid \tilde{B}^t\right]\end{aligned} \tag{26}$$

where $C = M\frac{(E_H^{\max})^2 + (\tilde{E}^{\max})^2}{2}$. The detailed proof of (26) is shown as follows.

Proof: We first introduce a statement. Let A, B, and C be the nonnegative real numbers and $W = A - B + C$, then $W^2 \leq A^2 + B^2 + C^2 + 2A(C - B)$. Recall the definition of battery level in (12), we have

$$\begin{aligned}\left(\tilde{b}_i^{t+1}\right)^2 &\leq \left(\tilde{b}_i^t\right)^2 + \left(\varepsilon_i^t\right)^2 + \left(e_i^t\right)^2 + 2\tilde{b}_i^t\left(e_i^t - \varepsilon_i^t\right) \\ &\leq \left(\tilde{b}_i^t\right)^2 + 2\tilde{b}_i^t\left(e_i^t - \varepsilon_i^t\right) + (E_H^{\max})^2 + \left(\tilde{E}^{\max}\right)^2.\end{aligned}$$

Reorganizing the abovementioned formula, we can obtain

$$\left(\tilde{b}_i^{t+1}\right)^2 - \left(\tilde{b}_i^t\right)^2 \leq 2\tilde{b}_i^t\left(e_i^t - \varepsilon_i^t\right) + (E_H^{\max})^2 + \left(\tilde{E}^{\max}\right)^2.$$

Summing all devices over time slot t, it holds

$$\begin{aligned}\sum_{i=1}^{M}\left[(\tilde{b}_i^{t+1})^2 - (\tilde{b}_i^t)^2\right] \leq\; & 2\sum_{i=1}^{M}\tilde{b}_i^t(e_i^t - \varepsilon_i^t) \\ & + M\left[(E_H^{\max})^2 + (\tilde{E}^{\max})^2\right].\end{aligned}$$

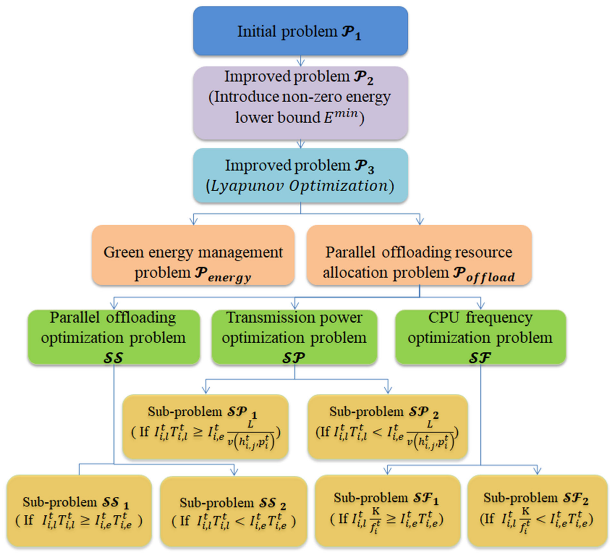
Initial problem $\mathcal{P}_1$
Improved problem $\mathcal{P}_2$ (Introduce non-zero energy lower bound E^{min})
Improved problem $\mathcal{P}_3$ (Lyapunov Optimization)
Green energy management problem $\mathcal{P}_{energy}$
Parallel offloading resource allocation problem $\mathcal{P}_{offload}$
Parallel offloading optimization problem $\mathcal{SS}$
Transmission power optimization problem $\mathcal{SP}$
CPU frequency optimization problem $\mathcal{SF}$
Sub-problem $\mathcal{SP}_1$ (If $I_{i,l}^t T_{i,l}^t \geq I_{i,e}^t \frac{L}{v(h_{i,j}^t, p_i^t)}$)
Sub-problem $\mathcal{SP}_2$ (If $I_{i,l}^t T_{i,l}^t < I_{i,e}^t \frac{L}{v(h_{i,j}^t, p_i^t)}$)
Sub-problem $\mathcal{SS}_1$ (If $I_{i,l}^t T_{i,l}^t \geq I_{i,e}^t T_{i,e}^t$)
Sub-problem $\mathcal{SS}_2$ (If $I_{i,l}^t T_{i,l}^t < I_{i,e}^t T_{i,e}^t$)
Sub-problem $\mathcal{SF}_1$ (If $I_{i,l}^t \frac{K}{f_i^t} \geq I_{i,e}^t T_{i,e}^t$)
Sub-problem $\mathcal{SF}_2$ (If $I_{i,l}^t \frac{K}{f_i^t} < I_{i,e}^t T_{i,e}^t$)

Fig. 2. Decoupling process of the problem.

By dividing both sides of the abovementioned inequality by 2, it gives

$$\Delta(t) \leq \sum_{i=1}^{M} \tilde{b}_i^t (e_i^t - \varepsilon_i^t) + M \left[\frac{\left(E_H^{\max}\right)^2 + \left(\tilde{E}^{\max}\right)^2}{2} \right].$$

Finally, by taking the expectation on the abovementioned inequality and adding the item $V\mathbb{E}[\text{cost}_{\text{total}}^t \mid \tilde{B}^t]$, it further yields (26). ∎

By scaling (25) with Lyapunov optimization, we embed the constraint on the stability of the energy queue into (26). Solving problem $\mathcal{P}_2$ is transformed into how to solve problem $\mathcal{P}_3$

$$\boldsymbol{\mathcal{P}_3}: \min \sum_{i=1}^{M} \left(\tilde{b}_i^t \left(e_i^t - \varepsilon_i^t \right) \right) + V \sum_{i=1}^{M} \left(D_i^t + \psi \chi_i^t \right) + C$$

$$\text{s.t.}: \quad (1)\text{–}(3),\ (8),\ (11),\ \text{and}\ (16)\text{–}(20).$$

Doing so not only minimizes the total system cost in a time-averaged sense, but also stabilizes the battery charge of each device.

B. Decoupling and Problem Solving

The problem $\mathcal{P}_3$ contains four variables to be determined: 1) green energy; 2) task division factor; 3) CPU frequency; and 4) transmission power. It is challenging to solve by traditional convex optimization algorithms. Our main idea is to decompose the problem $\mathcal{P}_3$ into a series of suboptimization problems at each time slot. In this part, we give the decoupling process of the problem theoretically and summarize it into Fig. 2.

1) First Decoupling of the Problem: We can find that the problem $\mathcal{P}_3$ can be decomposed into two subproblems: a) $\mathcal{P}_{\text{energy}}$; and 2) $\mathcal{P}_{\text{offload}}$. The former is to optimize the EH decision, that is, how to determine e_i^t, while the latter is to optimize parallel decision-making $S_i^t \triangleq [I_{i,l}^t, I_{i,e}^t]$, the CPU frequency $f_{i,k}^t$, and the transmission power p_i^t for resource allocation. We will give the optimal solution to the problem in each slot. Before discussing the problem further, we need to give a Lemma 2 [17] as follows.

Lemma 2: If the task is executed on the device side (locally) in the time slot t ($t \in \mathcal{T}$), the allocation of the CPU frequency will be optimal when K CPU cycles are equal, i.e., $f_{i,k}^t = f_i^t$, $i \in \mathcal{M}$, $k = 1, 2, \ldots, K$.

According to Lemma 2, we will use $T_{i,l}^t = K(f_i^t)^{-1}$ and $E_{i,l}^t = \theta K (f_i^t)^2, t \in \mathcal{T}, i \in \mathcal{M}$ to optimize the objective problem. For the energy optimization problem $\mathcal{P}_{\text{energy}}$, it is easy to obtain the optimal amount of EH e^{*t}_i by solving the following linear programming (LP) problem:

$$\mathcal{P}_{\text{energy}}: \quad \min \sum_{i=1}^{M} \tilde{b}_i^t e_i^t$$

$$\text{s.t.}: \quad 0 \leq e_i^t \leq E_H^t$$

$$e_i^t + b_i^t \leq \Lambda_i$$

where

$$e^{*t}_i = \begin{cases} \min\left\{\Lambda_i - b_i^t, E_H^t\right\}, & \tilde{b}_i^t \leq 0 \\ 0, & \tilde{b}_i^t > 0 \end{cases}. \tag{27}$$

Considering the remaining terms except for e_i^t in the problem $\mathcal{P}_3$, we can get the problem $\mathcal{P}_{\text{offload}}$ as follows:

$$\mathcal{P}_{\text{offload}}: \quad \min -\sum_{i=1}^{M} \tilde{b}_i^t \varepsilon_i^t + V \sum_{i=1}^{M} \left(D_i^t + \psi \chi_i^t \right) + C$$

$$\text{s.t.}: \quad (1)\text{–}(3),\ (11),\ \text{and}\ (16)\text{–}(20)$$

which includes three phases of operations in each time slot: 1) scheduling of CPU cycle frequency $f_{i,k}^t$; 2) distribution of transmission power p_i^t; and 3) determination of parallel offloading decision S_i^t. Since there are both continuous and discrete variables in the constraints, and the coupling between different variables is very high, it is still difficult to solve them directly. So, we try to decouple the problem a second time.

2) Second Decoupling of the Problem: Similar to [22], we convert $\mathcal{P}_{\text{offload}}$ into three equivalent subproblems in each time slot for the second decoupling. Taking the task division factor as the starting point, when the generated tasks are dropped, i.e., $I_{i,d}^t = 1$ and $I_{i,l}^t = I_{i,e}^t = 0$, the device neither needs to process the task nor send it to the edge server. Thus, we have $f_{i,k}^t = 0$ and $p_i^t = 0$. Next, we consider the case that the task is not dropped, and the following three equivalent subproblems can be obtained.

a) *Parallel Offloading Problem $\mathcal{SS}$:* When the transmission power and CPU frequency are given, i.e., $p_i^t = p_0^t$ and $f_i^t = f_0^t$, respectively, we can get the optimal solution S^{*t}_i.

$$\mathcal{SS}: \quad \min_{S_i^t} -\tilde{b}_i^t \left(I_{i,1}^t E_{i,l}^t + I_{i,e}^t E_{i,e}^t \right)$$

$$+ V \cdot \max\left\{ I_{i,1}^t T_{i,l}^t, I_{i,e}^t T_{i,e}^t \right\}$$

$$\text{s.t.}: (1),\ (2),\ (16), \text{and}\ (20).$$

b) *Transmission Power Problem $\mathcal{SP}$:* When parallel offloading decision and CPU frequency are given, i.e., $S_i^t = S_0^t$

and $f_i^t = f_0^t$, respectively, the optimal solution p^{*t}_i can be obtained.

$$\mathcal{SP}: \min_{p_i^t} -\tilde{b}_i^t \left[I_{i,l}^t E_{i,l}^t + I_{i,e}^t \frac{p_i^t L}{\omega \log_2 \left(1 + \frac{h_{i,j}^t p_i^t}{\sigma}\right)} \right]$$
$$+ V \cdot \max \left\{ I_{i,l}^t T_{i,l}^t, I_{i,e}^t \frac{L}{\omega \log_2 \left(1 + \frac{h_{i,j}^t p_i^t}{\sigma}\right)} \right\}$$
$$\text{s.t.}: (16) \text{and } (19)$$
$$I_{i,e}^t E_{i,e}^t \in \left[\max\left\{0, E^{\min} - I_{i,l}^t E_{i,l}^t\right\}, E^{\max} - I_{i,l}^t E_{i,l}^t\right].$$

c) *CPU Frequency Problem $\mathcal{SF}$:* When parallel offloading decision and transmission power are given, i.e., $S_i^t = S_0^t$ and $p_i^t = p_0^t$, respectively, the optimal solution f^{*t}_i can be obtained.

$$\mathcal{SF}: \min_{f_i^t} -\tilde{b}_i^t \left(\theta I_{i,l}^t K \left(f_i^t\right)^2 + I_{i,e}^t E_{i,e}^t\right)$$
$$+ V \cdot \max \left\{ I_{i,l}^t \frac{K}{f_i^t}, I_{i,e}^t T_{i,e}^t \right\}$$
$$\text{s.t.}: (16) \text{and} (18)$$
$$I_{i,l}^t E_{i,l}^t \in \left[\max\left\{0, E^{\min} - I_{i,e}^t E_{i,e}^t\right\}, E^{\max} - I_{i,e}^t E_{i,e}^t\right].$$

3) Third Decoupling of the Problem: We found that the way, in which tasks are offloaded in parallel makes D_i^t [in (13)] difficult to solve on the three subproblems. Here, we take D_i^t as the starting point to decouple the abovementioned series of problems for the third time and give the expression of the optimal solution.

The parallel offloading problem $\mathcal{SS}$ is a convex optimization problem about the variable S_i^t, which consists of several convex functions added together and can be further transformed into subproblems $\mathcal{SS}_1$ and $\mathcal{SS}_2$.

a) *Subproblem $\mathcal{SS}_1$ for case $I_{i,l}^t T_{i,l}^t \geq I_{i,e}^t T_{i,e}^t$:*

$$\mathcal{SS}_1: \min_{S_i^t} -\tilde{b}_i^t \left(I_{i,1}^t E_{i,l}^t + I_{i,e}^t E_{i,e}^t\right) + V I_{i,1}^t T_{i,l}^t$$
$$\text{s.t.}: \quad (1),\ (2),\ (16),\ \text{and } (20).$$

b) *Subproblem $\mathcal{SS}_2$ for case $I_{i,l}^t T_{i,l}^t < I_{i,e}^t T_{i,e}^t$:*

$$\mathcal{SS}_2: \min_{S_i^t} -\tilde{b}_i^t \left(I_{i,1}^t E_{i,l}^t + I_{i,e}^t E_{i,e}^t\right) + V I_{i,e}^t T_{i,e}^t$$
$$\text{s.t.}: \quad (1),\ (2),\ (16),\ \text{and } (20).$$

We can use LP tools to obtain the optimal solution for each problem easily, and apply the contradiction method to verify whether the result meets the assumptions, so as to obtain the optimal offloading decision S^{*t}_i.

The transmission power problem $\mathcal{SP}$ can be further reduced to subproblems $\mathcal{SP}_1$ and $\mathcal{SP}_2$ as follows.

a) *Subproblem $\mathcal{SP}_1$ for case $I_{i,l}^t T_{i,l}^t \geq I_{i,e}^t \frac{L, v(h_{i,j}^t, p_i^t)}{\omega h_{i,j}^t}$:*

$$\mathcal{SP}_1: \min_{p_i^t} -\tilde{b}_i^t \left[I_{i,l}^t E_{i,l}^t + I_{i,e}^t \frac{p_i^t L}{v\left(h_{i,j}^t, p_i^t\right)} \right] + V I_{i,l}^t T_{i,l}^t$$
$$\text{s.t.}: 0 \leq p_i^t \leq p^{\max}$$
$$I_{i,e}^t \frac{L}{v\left(h_{i,j}^t, p_i^t\right)} \leq I_{i,l}^t T_{i,l}^t \leq \tau$$
$$I_{i,e}^t E_{i,e}^t \in \left[\max\left\{0, E^{\min} - I_{i,l}^t E_{i,l}^t\right\}, E^{\max} - I_{i,l}^t E_{i,l}^t\right]$$

where $\tau' = I_{i,l}^t T_{i,l}^t$ in this case, and

$$p^{*t}_i = \begin{cases} p_U, & \tilde{b}_i^t \geq 0 \\ p_L, & \tilde{b}_i^t < 0 \end{cases}. \tag{28}$$

b) *Subproblem $\mathcal{SP}_2$ for case $I_{i,l}^t T_{i,l}^t < I_{i,e}^t \frac{L, v(h_{i,j}^t, p_i^t)}{\omega h_{i,j}^t}$:*

$$\mathcal{SP}_2: \min_{p_i^t} -\tilde{b}_i^t \left[I_{i,l}^t E_{i,l}^t + I_{i,e}^t \frac{p_i^t L}{v\left(h_{i,j}^t, p_i^t\right)} \right]$$
$$+ V I_{i,e}^t \frac{L}{v\left(h_{i,j}^t, p_i^t\right)}$$
$$\text{s.t.}: I_{i,e}^t \frac{L}{v\left(h_{i,j}^t, p_i^t\right)} \leq \tau$$
$$0 \leq p_i^t \leq p^{\max}$$
$$I_{i,e}^t E_{i,e}^t \in \left[\max\left\{0, E^{\min} - I_{i,l}^t E_{i,l}^t\right\}, E^{\max} - I_{i,l}^t E_{i,l}^t\right]$$

where $\tau' = \tau$ in this case, and

$$p^{*t}_i = \begin{cases} p_U, & \tilde{b}_i^t \geq 0 \text{ or } \tilde{b}_i^t < 0\ \&\ p_0 > p_U \\ p_0, & \tilde{b}_i^t < 0\ \&\ p_L < p_0 < p_U \\ p_L, & \tilde{b}_i^t < 0\ \&\ p_0 < p_L \end{cases}. \tag{29}$$

Let $\mathrm{g}_1(p_i^t, h_{i,j}^t, \tilde{b}_i^t) = \frac{-\tilde{b}_i^t p_i^t}{v(h_{i,j}^t, p_i^t)} + \frac{V}{v(h_{i,j}^t, p_i^t)}$, we take the first-order partial derivative of g_1: $\frac{\mathrm{dg}_1(p_i^t, h_{i,j}^t, \tilde{b}_i^t)}{\mathrm{d}p_i^t} = \frac{-\tilde{b}_i^t \log_2(1+\frac{h_{i,j}^t p_i^t}{\sigma}) - \frac{h_{i,j}^t}{(\sigma + h_{i,j}^t p_i^t)\ln 2}(V - p_i^t \tilde{b}_i^t)}{\omega \log_2^2(1+\frac{h_{i,j}^t p_i^t}{\sigma})}$ and p_0 is the solution of $\tilde{b}_i^t \log_2(1 + \frac{h_{i,j}^t p_i^t}{\sigma}) + \frac{h_{i,j}^t}{(\sigma + h_{i,j}^t p_i^t)\ln 2}(V - p_i^t \tilde{b}_i^t) = 0$. In addition, we define p_L and p_U as follows:

$$p_L = \begin{cases} p_{L,\tau'}, & E_0 \geq E^{\min} - I_{i,l}^t E_{i,l}^t \\ \max\left\{p_{L,\tau'}, p_{E^{\min}}\right\}, & E_0 < E^{\min} - I_{i,l}^t E_{i,l}^t \end{cases} \tag{30}$$

$$p_U = \begin{cases} 0, & E_0 \geq E^{\max} - I_{i,l}^t E_{i,l}^t \\ \min\left\{p^{\max}, p_{E^{\max}}\right\}, & E_0 < E^{\max} - I_{i,l}^t E_{i,l}^t \end{cases} \tag{31}$$

where $E_0 = \frac{\sigma I_{i,e}^t L \ln 2}{\omega h_{i,j}^t}$ and $p_{L,\tau'} = \frac{\sigma(2^{\frac{L I_{i,e}}{\omega \tau'}} - 1)}{h_{i,j}^t}$ is the solution, when $T_{i,e}^t = \tau'$. Besides, $p_{E^{\min}}$ and $p_{E^{\max}}$ are the solutions of $I_{i,e}^t E_{i,e}^t = E^{\min}$ and $I_{i,e}^t E_{i,e}^t = E^{\max}$, respectively. That is, $p_{E^{\min}} I_{i,e}^t L = v(h_{i,j}^t, p_{E^{\min}}) E^{\min}$ and $p_{E^{\max}} I_{i,e}^t L = v(h_{i,j}^t, p_{E^{\max}}) E^{\max}$.

Furthermore, the CPU frequency allocation problem $\mathcal{SF}$ can be further reduced to subproblems $\mathcal{SF}_1$ and $\mathcal{SF}_2$.

a) *Subproblem $\mathcal{SF}_1$ for case $I_{i,l}^t \frac{K}{f_i^t} \geq I_{i,e}^t T_{i,e}^t$:*

$$\mathcal{SF}_1: \min_{f_i^t} -\tilde{b}_i^t \left[I_{i,l}^t \theta K \left(f_i^t\right)^2 + I_{i,e}^t E_{i,e}^t \right] + V I_{i,l}^t \frac{K}{f_i^t}$$
$$\text{s.t.}: I_{i,l}^t \frac{K}{f_i^t} \leq \tau$$

$$0 \le f_i^t \le f_{\text{local}}^{\max}$$

$$I_{i,l}^t E_{i,l}^t \in \left[\max\left\{0, E^{\min} - I_{i,e}^t E_{i,e}^t\right\}, E^{\max} - I_{i,e}^t E_{i,e}^t\right]$$

where $\tau' = \tau$ in this case, and

$$f^{*t}_i = \begin{cases} f_U, & \tilde{b}_i^t \ge 0 \text{ or } \tilde{b}_i^t < 0, f_0 > f_U \\ f_0, & \tilde{b}_i^t < 0, f_L < f_0 < f_U \\ f_L, & \tilde{b}_i^t < 0, f_0 < f_L \end{cases}. \tag{32}$$

Let $\mathrm{g}_2(f_i^t, \tilde{b}_i^t) = -\tilde{b}_i^t \theta K (f_i^t)^2 + V\frac{K}{f_i^t}$, we take the first-order partial derivative of g_2: $\frac{\mathrm{d}\mathrm{g}_2(f_i^t,\tilde{b}_i^t)}{\mathrm{d}p_i^t} = -2\tilde{b}_i^t\theta K f_i^t - V\frac{K}{f_i^t}$ and f_0 is the solution of $\mathrm{g}_3(f_i^t, \tilde{b}_i^t)$, i.e., $f_0 = \sqrt[3]{\frac{V}{-2\theta\tilde{b}_i^t}}$. Besides, $f_L = \max\left\{\frac{KI_{i,l}^t}{\tau'}, \max\left\{0, \sqrt{\frac{E^{\min} - I_{i,e}^t E_{i,e}^t}{I_{i,l}^t \theta K}}\right\}\right\}$ and $f_U = \min\left\{f_{\text{local}}^{\max}, \sqrt{\frac{E^{\max} - I_{i,e}^t E_{i,e}^t}{I_{i,l}^t \theta K}}\right\}$.

b) *Subproblem $\mathcal{SF}_2$ for case $I_{i,l}^t \frac{K}{f_i^t} < I_{i,e}^t T_{i,e}^t$:*

$$\mathcal{SF}_2 : \min_{f_i^t} -\tilde{b}_i^t\left(I_{i,l}^t \theta K \left(f_i^t\right)^2 + I_{i,e}^t E_{i,e}^t\right) + V I_{i,e}^t T_{i,e}^t$$

$$\text{s.t.}: I_{i,l}^t \frac{K}{f_i^t} < I_{i,e}^t T_{i,e}^t \le \tau$$

$$0 \le f_i^t \le f_{\text{local}}^{\max}$$

$$I_{i,l}^t E_{i,l}^t \in \left[\max\left\{0, E^{\min} - I_{i,e}^t E_{i,e}^t\right\}, E^{\max} - I_{i,e}^t E_{i,e}^t\right]$$

where $\tau' = I_{i,e}^t T_{i,e}^t$ in this case, and

$$f^{*t}_i = \begin{cases} f_U, & \tilde{b}_i^t \ge 0 \\ f_L, & \tilde{b}_i^t < 0 \end{cases}. \tag{33}$$

C. Green Parallel Online Offloading Algorithm

As we mentioned earlier, the computing resources of edge servers in the real world are usually limited. We assume that MEC servers in the IoT-Edge system have limited computing resources, that is, in each time slot, at most $Q = [\frac{\tau f_{\text{edge}}^{\max}}{K}]$ devices are allowed to connect to one MEC server at the same time. According to the optimal solution given in Section III, we design the GPOOA algorithm for DSCI-type tasks.

GPOOA adopts the principle of minimum target value (J_{parallel}^t) first offloading, and seeks for parallel decision-making and resource allocation scheme to minimize the system cost. It is known that when given any two variables, the $\mathcal{P}_3$ turns into how to optimize $\mathcal{SF}$, $\mathcal{SP}$, or $\mathcal{SS}$. First, we get a random task division factor $\mathcal{S}_{i,j}^t$ by initializing $f_{i,j(0)}^t$ and $p_{i,j(0)}^t$, and get a random $p_{i,j}^t$ by $\mathcal{S}_{i,j}^t$ and $f_{i,j(0)}^t$. We use the obtained $\mathcal{S}_{i,j}^t$ and $p_{i,j}^t$ as the given two variables to optimize the problem $\mathcal{SF}$. Meanwhile, if the optimal offload object of task A generated by the device i is the MEC server j and the number of devices connected to server j is less than Q, then task A will be completed by the device i and the server j together. If the server j has connected Q devices, then task A can only choose the suboptimal offload object. The details of the algorithmic process are described in Algorithm 1, where $J_{\text{parallel}}^t = -\tilde{b}_i^t \varepsilon_i^t + V(D_i^t + \psi\chi_i^t)$ and Q is the maximum number of devices that the edge server can connect to in a time slot. Fig. 3 shows the DSCI-type task division process.

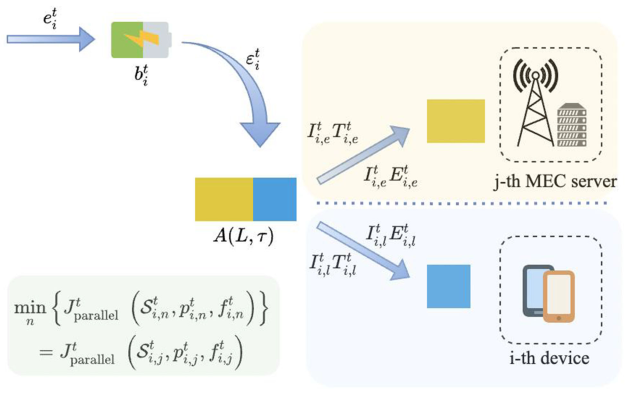
e_i^t
b_i^t
ε_i^t
$I_{i,e}^t T_{i,e}^t$
$I_{i,e}^t E_{i,e}^t$
j-th MEC server
$A(L,\tau)$
$I_{i,l}^t E_{i,l}^t$
$I_{i,l}^t T_{i,l}^t$
$\min_n\left\{J_{\text{parallel}}^t\left(\mathcal{S}_{i,n}^t, p_{i,n}^t, f_{i,n}^t\right)\right\} = J_{\text{parallel}}^t\left(\mathcal{S}_{i,j}^t, p_{i,j}^t, f_{i,j}^t\right)$
i-th device

Fig. 3. Task division process for the DSCI-type tasks.

IV. Performance Evaluation

In this section, we verify the effectiveness of GPOOA through MATLAB simulation with the adoption of the controlled variable method.

A. Simulation Setup

The parameter setting of this article mainly refers to works in [13] and[17]. There are three MEC servers and eight IoT devices placed in an area of 100×100 m, where IoT devices can move arbitrarily in the area without affecting each other. Let $E_{i,H}^t$ be a uniform distribution on $[0, E_H^{\max}]$ with the average EH power $p_H = E_H^{\max}/2\tau$ (the range is between 7.5 and 10 mW). The unit task $A(L,\tau)$ with $L = 1$ kbits and $\tau = 2$ ms. The channel power gains are exponential distribution with mean $g_0(\frac{d_0}{d_{i,j}^t})^\alpha$, where the pass-loss exponent $\alpha = 4$, the path-loss constant $g_0 = -40$ dB, and $d_0 = 1$. The small-scale fading channel power gains follow an exponential distribution, i.e., $\gamma_{i,j}^t \sim \text{Exp}(1)$. In addition, $\theta = 10^{-28}$, $f_{\text{local}}^{\max} = 1.5$ GHz, $p^{\max} = 1.8$ W, $\omega = 10^6$ Hz, and $\sigma = 10^{-13}$. Penalty weight for dropping tasks cost $\psi = 2$ ms, $E^{\min} = 0.04$ mJ, $f_{\text{edge}}^{\max} = 1.5$ GHz, and $W = 737.5$ cycle/bit. We verify the effectiveness of GPOOA through MATLAB simulation on 3000 time slots with the slot length $\tau_0 = 2$ ms.

B. Performance Analysis

As depicted in Fig. 4, given the arrival rate $\rho = 0.5$, as V goes from 0 to 7×10^{-5}, the time-averaged system cost drops from 1.57 to 0.97 ms and the average energy queue backlog increases from 1.48 to 2.70 mJ. It can be found that the average battery queue length increases linearly as V increases. Meanwhile, the system cost is inversely proportional to V and eventually converges to the optimal value of $\mathcal{P}_1$ as V increases. Thus, by adjusting V, the tradeoff between the minimization of the system cost and the stability of the battery queue can be achieved.

From Fig. 5, the system cost decreases rapidly at the beginning, then tends to decrease slowly, and finally stays within

Algorithm 1: GPOOA algorithm.

```
1:  for time slots $t \in \mathcal{T}$ do
2:    for $i = 1$ to $M$ do
3:      Acquire $\zeta_i^t$, $\tilde{b}_i^t$ and $E_H^t$
4:      Solve the problem $\mathcal{P}_{\text{energy}}$ as (27) to get the $e^{*t}_i$
5:      for $j = 1$ to $N$ do
6:        Initialize $f_{i,j(0)}^t$ and $p_{i,j(0)}^t$
7:        Solve the problem $\mathcal{SS}$ to get the $\mathcal{S}_{i,j}^t$;
8:        Solve the problem $\mathcal{SP}$ as Eqs.(28) and (29) to
          get the $p_{i,j}^t$;
9:        Solve the problem $\mathcal{SF}$ as Eqs.(32) and (33) to
          get the $f_{i,j}^t$;
10:       Record optimal value $J_{\text{parallel}}^t(\mathcal{S}_{i,j}^t, p_{i,j}^t, f_{i,j}^t)$;
11:       If the battery energy level is insufficient for the $i$
          IoT device and the $j$ MEC server to parallel
          offloading, set $J_{\text{parallel}}^t(\mathcal{S}_{i,j}^t, p_{i,j}^t, f_{i,j}^t)$ as inf;
12:       Choose the optimal $\mathcal{S}^{*t}_i$, $P^{*t}_i$ and $f^{*t}_i$ by
          selecting the minimum $J_{\text{parallel}}^t(\mathcal{S}_{i,j}^t, p_{i,j}^t, f_{i,j}^t)$,
          denote as $J_{\text{parallel}}^t(\mathcal{S}_i^t, p_i^t, f_i^t)$ and record $j$.
13:     end for
14:     Insert key-value pair into the map with key $i$ and
        value $j$;
15:   end for
16:   while map $\neq \emptyset$ do
17:     Find the key-value pair"$i$–$j$" with the smallest
        value $J_{\text{parallel}}^t(\mathcal{S}_i^t, p_i^t, f_i^t)$ and record $j$;
18:     if flag[$j$] $\leq Q$ then
19:       Remove the key-value pair "$i$–$j$" from the map;
20:       flag[$j$] = flag[$j$] + 1;
21:     else
22:       $J_{\text{parallel}}^t(\mathcal{S}_{:,j}^t, p_{:,j}^t, f_{:,j}^t) = \text{inf}$;
23:       if $\min\{J_{\text{parallel}}^t(\mathcal{S}_{i,:}^t, p_{i,:}^t, f_{i,:}^t)\} \neq \text{inf}$ then
24:         Find the smallest $J_{\text{parallel}}^t(\mathcal{S}_{i,j'}^t, p_{i,j'}^t, f_{i,j'}^t)$,
            overwrite the server initially selected in the
            map with $j'$, and overwrite the value of
            $J_{\text{parallel}}^t(\mathcal{S}_i^t, p_i^t, f_i^t)$.
25:       else
26:         There is no server to choose, the task can only
            be dropped.
27:       end if
28:     end if
29:   end while
30:   Update the virtual energy queue $\tilde{b}_i^{t+1}$;
31:   Set $t = t + 1$;
32: end for
```

1.1 ms. As time increases, the energy buffer of the battery queue gradually increases and remains at a relatively stable level after the 1000th time slot. That is, the green energy captured by the outside world and the energy consumed by the task have reached a balanced state. We can also find from the curve that the battery queue backlog fluctuates frequently. This is because when the energy harvester captures energy from the outside world, the process of green energy reaching the device at time slot t is random.

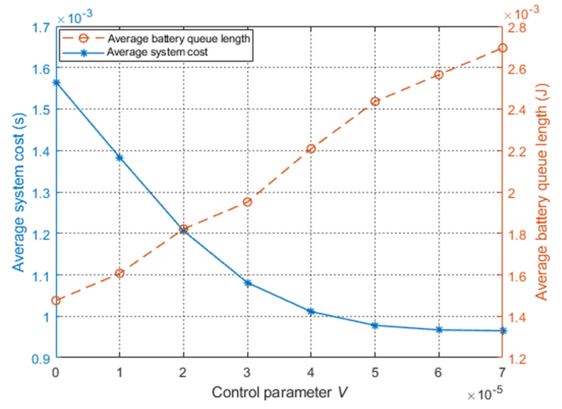

Fig. 4. Average system cost and average battery queue level under different control parameters V (in $J^2 \cdot s^{-1}$).

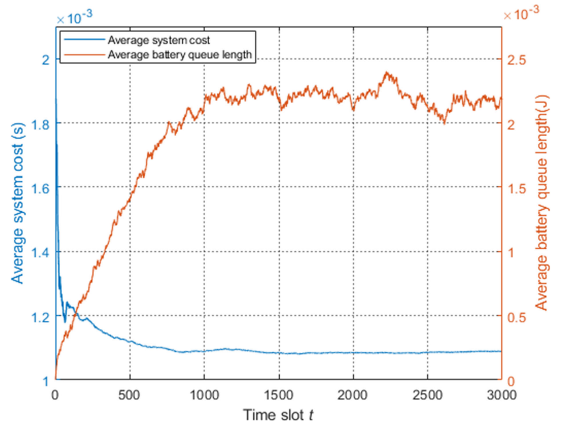

Fig. 5. Average system cost and the average battery queue length over 3000 time slots with the arrival rate $\rho = 0.5$ and $V = 3 \times 10^{-5}$.

From Fig. 6, the task drop ratio tends to be very small after the 500th time slot, which indicates that most of the generated tasks can be completed within the deadline. This is because the GPOOA utilizes a combination of green energy to provide power and parallel offloading. Parallel offloading reduces the execution time of tasks and ensures that they are processed within the deadline as much as possible. Green energy solves the problem of limited local batteries and provides a constant source of energy support for smooth parallel offloading of tasks, which results in a significant reduction in task drop ratio.

C. Comparison of Different Offloading Schemes

We compare the proposed GPOOA with the following three offloading methods: 1) LODCO [17]; 2) the only-edge algorithm (OEA); and 3) DOA. These algorithms are all performed on 3000 time slots. The devices are powered by the green energy collected by the EH. Tasks are generated with a Bernoulli distribution and are of the DSCI type.

i) *Only-Edge Algorithm:* This algorithm adopts a method in which tasks are greedily offloaded to edge servers

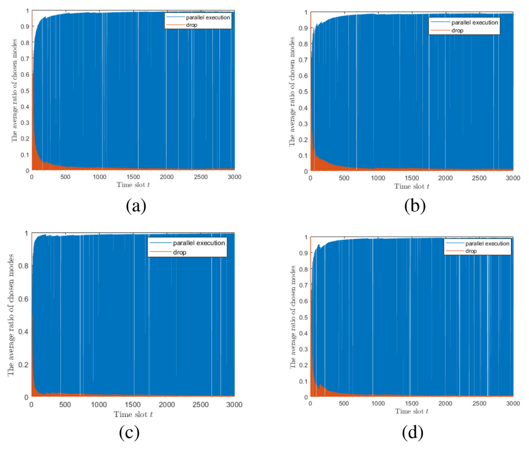

Fig. 6. Evolution of the average task drop ratio of devices (orange part), where the first, third, fifth, and seventh devices are selected in turn. (a) First device. (b) Third device. (c) Fifth device. (d) Seventh device.

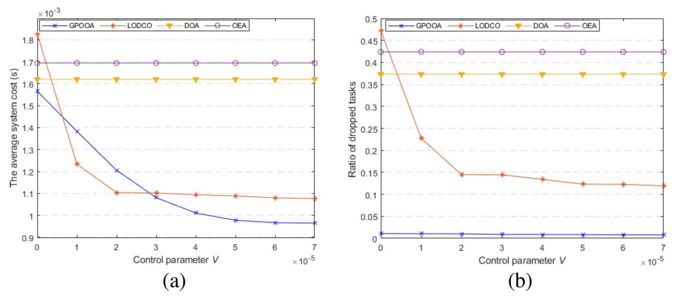

Fig. 7. Average system cost and the ratio of dropped tasks under different control parameters V. (a) Average system cost. (b) Ratio of dropped tasks.

for processing. When a) $\frac{L}{v(h_{i,j}^t, p_i^t)} \leq \tau$; and b) the energy required for the transmission task does not exceed the battery energy level of the current time slot, the task will be offloaded with the maximum transmission power p_i^t. Here, $p_i^t = \min\{p^{\max}, p^t_{\min\{b^t, E^{\max}\}}\}$, if $\frac{\sigma L \ln 2}{\omega h^t} < \min\{b_i^t, E^{\max}\}$ and $p^t_{\min\{b_i^t, E^{\max}\}}$ is the unique solution of $pL = v(h_{i,j}^t, p)\min\{b_i^t, E^{\max}\}$. Otherwise, the task will be dropped.

ii) *Dynamic Offloading Algorithm:* This algorithm adopts a calculation mode with a smaller delay within the delay requirement ($\frac{L}{v(h_{i,j}^t, p_i^t)} \leq \tau$ or $\frac{W}{f_i^t} \leq \tau$). That is, if the delay due to offloading is less than the local execution time, the task will be offloaded to the edge with the maximum transmission power p_i^t. Otherwise, it will be processed locally with the maximum CPU cycle frequency f_i^t. If neither of these two calculation modes is feasible, the task will be dropped. The maximum CPU cycle frequency available on the device side $f_i^t = \min\{f_{\text{local}}^{\max}, \sqrt{\frac{\min\{b_i^t, E^{\max}\}}{\theta W}}\}$ if $\frac{W}{f_i^t} \leq \tau$. The maximum transmission power p_i^t is the same as the OEA algorithm.

1) Impact of the Control Parameter V: It can be observed from Fig. 7(a) that our proposed GPOOA performs better in

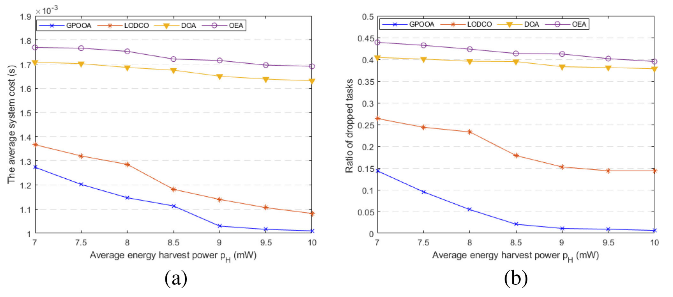

Fig. 8. Average system cost and the ratio of dropped tasks versus p_H, where $V = 4 \times 10^{-5}$. (a) Average system cost. (b) Ratio of dropped tasks.

minimizing system cost with V increases and outperforms the other three algorithms. As V increases, the system cost of the LODCO and the GPOOA decreases by $O(1/V)$ and gradually converges to the lowest level. This is because both the LODCO and the GPOOA are based on the Lyapunov optimization framework. In addition, this also confirms that GPOOA can reach the asymptotic optimum. Since DOA and OEA do not require V to regulate system costs and battery queues, their average system costs do not change with changes in V. At the same time, we found that the DOA is better than the OEA, which only greedily offloads to the edge. As we envisioned, only greedy offloading on the device-side or edge-side cannot make full use of resources.

2) Impact of the Task Drop Ratio: From Fig. 7, the inability of DOA and OEA to make full use of system resources has resulted in large delays and high energy consumption. This caused a large number of tasks to be dropped due to excessive response time or insufficient energy supply. With the increase of V, LODCO suppresses the task drop ratio at the expense of a smaller system cost drop. However, compared with the other three schemes, the proposed GPOOA has outstanding performance in reducing the task drop ratio. Because GPOOA can flexibly select edge servers to parallel offloading for each IoT device based on the current channel status and user location. Not only does it take full advantage of the green energy on the device side, but it also takes advantage of the computing power of the edge server, so that more tasks can be executed within the deadline.

3) Impact of the EH Rate: Fig. 8(a) and (b) show the relationship between the average system cost and the task drop ratio with p_H. With the increase of p_H, the average system cost and task drop ratio of all algorithms have decreased to varying degrees. Because the consumption of green energy does not incur system costs, the larger the p_H, the more sufficient energy is provided for the device in a unit time slot. Adequate energy will weaken the energy consumption constraints caused by offloading, so more tasks will be executed smoothly. Our GPOOA has outstanding performance in task drop ratio, which is less than 1% after $p_H = 8.5$ mW. The average system cost of GPOOA is much lower than that of the DOA and the OEA. Compared with the LODCO, the average system cost is reduced by 8.21%. This once again verifies the effectiveness of the GPOOA algorithm.

V. Conclusion

This article proposed GPOOA, a green parallel offloading strategy based on Lyapunov optimization. Through three-time decoupling of the objective function, the joint optimization of green energy, CPU cycle frequency, transmission power, and task division factor was realized. Under the condition of little prior knowledge, the algorithm flexibly selected the target server for parallel offloading according to the current location and channel state of the device. In addition, we conducted a performance analysis and revealed that GPOOA can achieve asymptotically optimal results. Simulation results demonstrated that through parallel offloading, GPOOA was significantly superior to benchmark strategies in terms of system costs and task drop ratio. Future work includes applying Lyapunov-guided federated reinforcement learning [28] and digital twin [29] to simulate the EH process.

References

[1] S. Painuly, S. Sharma, and P. Matta, "Future trends and challenges in next generation smart application of 5G-IoT," in *Proc. 5th Int. Conf. Comput. Methodol. Commun.*, 2021, pp. 354–357.

[2] Y. Chen, S. Zhang, M. Xiao, Z. Qian, J. Wu, and S. Lu, "Multi-user edge-assisted video analytics task offloading game based on deep reinforcement learning," in *Proc. IEEE 26th Int. Conf. Parallel Distrib. Syst.*, 2020, pp. 266–273.

[3] I. Attiya, M. A. Elaziz, L. Abualigah, T. N. Nguyen, and A. A. A. El-Latif, "An improved hybrid swarm intelligence for scheduling IoT application tasks in the cloud," *IEEE Trans. Ind. Informat.*, to be published, doi: 10.1109/TII.2022.3148288.

[4] L. Zhao, K. Yang, Z. Tan, X. Li, S. Sharma, and Z. Liu, "A novel cost optimization strategy for SDN-enabled UAV-assisted vehicular computation offloading," *IEEE Trans. Intell. Transp. Syst.*, vol. 22, no. 6, pp. 3664–3674, Jun. 2021.

[5] Y. Li, Z. Han, Q. Zhang, Z. Li, and H. Tan, "Automating cloud deployment for deep learning inference of real-time online services," in *Proc. IEEE Conf. Comput. Commun.*, 2020, pp. 1668–1677.

[6] J. Xu, L. Chen, and S. Ren, "Online learning for offloading and autoscaling in energy harvesting mobile edge computing," *IEEE Trans. Cogn. Commun. Netw.*, vol. 3, no. 3, pp. 361–373, Sep. 2017.

[7] G. Qu, N. Cui, H. Wu, R. Li, and Y. Ding, "ChainFL: A simulation platform for joint federated learning and blockchain in edge/cloud computing environments," *IEEE Trans. Ind. Informat.*, vol. 18, no. 5, pp. 3572–3581, May 2022.

[8] A. A. Kherani *et al.*, "Development of MEC system for indigenous 5G test-bed," in *Proc. Int. Conf. Commun. Syst. Netw.*, 2021, pp. 131–133.

[9] K. Peng, H. Huang, B. Zhao, A. Jolfaei, X. Xu, and M. Bilal, "Intelligent computation offloading and resource allocation in IIoT with end-edge-cloud computing using NSGA-III," *IEEE Trans. Netw. Sci. Eng.*, to be published, doi: 10.1109/TNSE.2022.3155490.

[10] P. Brown, "Is battery life hindering the growth of Internet of Things devices?," Accessed: Nov. 5, 2018. [Online]. Available: https://electronics360.globalspec.com/article/13112/is-battery-life-hindering-the-growth-of-internet-of-things-devices

[11] Z. Zhou, Z. Chang, and H. Liao, "Dynamic computation offloading scheme for fog computing system with energy harvesting devices," in *Green Internet of Things (IoT): Energy Efficiency Perspective*, Cham, Switzerland: Springer, 2021, pp. 143–161.

[12] A. Sharma and P. Sharma, "Energy harvesting technology for IoT edge applications," in *Smart Manufacturing*, T. Y. Kheng, Ed. London, U.K.: IntechOpen, 2021. [Online]. Available: https://doi.org/10.5772/intechopen.92565

[13] H. Zhao, W. Du, W. Liu, T. Lei, and Q. Lei, "QoE aware and cell capacity enhanced computation offloading for multi-server mobile edge computing systems with energy harvesting devices," in *Proc. IEEE SmartWorld, Ubiquitous Intell. Comput., Adv. Trusted Comput., Scalable Comput. Commun., Cloud Big Data Comput., Internet People Smart City Innov.*, 2018, pp. 671–678.

[14] M. Guo, L. Li, and Q. Guan, "Energy-efficient and delay-guaranteed workload allocation in IoT-edge-cloud computing systems," *IEEE Access*, vol. 7, pp. 78685–78697, 2019.

[15] H. Wu, K. Wolter, P. Jiao, Y. Deng, Y. Zhao, and M. Xu, "EEDTO: An energy-efficient dynamic task offloading algorithm for blockchain-enabled IoT-edge-cloud orchestrated computing," *IEEE Internet Things J.*, vol. 8, no. 4, pp. 2163–2176, Feb. 2021.

[16] Y. Chen, N. Zhang, Y. Zhang, X. Chen, W. Wu, and X. Shen, "Energy efficient dynamic offloading in mobile edge computing for Internet of Things," *IEEE Trans. Cloud Comput.*, vol. 9, no. 3, pp. 1050–1060, Jul.–Sep. 2021.

[17] 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.

[18] A. Yousefpour, G. Ishigaki, and J. P. Jue, "Fog computing: Towards minimizing delay in the Internet of Things," in *Proc. IEEE Int. Conf. Edge Comput.*, 2017, pp. 17–24.

[19] M. Mukherjee, S. Kumar, M. Shojafar, Q. Zhang, and C. X. Mavromoustakis, "Joint task offloading and resource allocation for delay-sensitive fog networks," in *Proc. IEEE Int. Conf. Commun.*, 2019, pp. 1–7.

[20] 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*, 2016, pp. 1451–1455.

[21] Z. Zhou, "GreenEdge: Greening edge datacenters with energy-harvesting IoT devices," in *Proc. IEEE 27th Int. Conf. Netw. Protoc.*, 2019, pp. 1–6.

[22] Y. Deng, Z. Chen, X. Yao, S. Hassan, and A. M. A. Ibrahim, "Parallel offloading in green and sustainable mobile edge computing for delay-constrained IoT system," *IEEE Trans. Veh. Technol.*, vol. 68, no. 12, pp. 12202–12214, Dec. 2019.

[23] H. Hu, Q. Wang, R. Q. Hu, and H. Zhu, "Mobility-aware offloading and resource allocation in a MEC-enabled IoT network with energy harvesting," *IEEE Internet Things J.*, vol. 8, no. 24, pp. 17541–17556, Dec. 2021.

[24] T. Robertazzi, "Ten reasons to use divisible load theory," *Computer*, vol. 36, no. 5, pp. 63–68, May, 2003.

[25] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surv. Tut.*, vol. 19, no. 4, pp. 2322–2358, Oct.–Dec. 2017.

[26] T. Liu, Y. Zhang, Y. Zhu, W. Tong, and Y. Yang, "Online computation offloading and resource scheduling in mobile-edge computing," *IEEE Internet Things J.*, vol. 8, no. 8, pp. 6649–6664, Apr. 2021.

[27] Z. Chang, L. Liu, X. Guo, and Q. Sheng, "Dynamic resource allocation and computation offloading for IoT fog computing system," *IEEE Trans. Ind. Informat.*, vol. 17, no. 5, pp. 3348–3357, May 2021.

[28] X. Wang *et al.*, "QoS and privacy-aware routing for 5G enabled industrial Internet of Things: A federated reinforcement learning approach," *IEEE Trans. Ind. Informat.*, vol. 18, no. 6, pp. 4189–4197, Jun. 2022.

[29] Y. Dai, K. Zhang, S. Maharjan, and Y. Zhang, "Deep reinforcement learning for stochastic computation offloading in digital twin networks," *IEEE Trans. Ind. Informat.*, vol. 17, no. 7, pp. 4968–4977, Jul. 2021.

Junqi Chen received the B.S. degree in mathematics from the Taiyuan University of Technology, Taiyuan, China, in 2019. She is currently working toward the master's degree in mathematics with the Center for Applied Mathematics, Tianjin University, Tianjin, China.

Her research interests include Internet of Things, deep learning, and mobile edge computing.

Huaming Wu (Senior Member, IEEE) received the B.E. and M.S. degrees in electrical engineering from the Harbin Institute of Technology, Harbin, China, in 2009 and 2011, respectively, the Ph.D. degree in computer science from Freie Universität Berlin, Berlin, Germany, in 2015.

He is currently an Associate Professor with the Center for Applied Mathematics, Tianjin University, Tianjin, China. His research interests include wireless networks, mobile edge computing, Internet of Things, and deep learning.

Ruidong Li (Senior Member, IEEE) received the M.Sc. and Ph.D. degrees in computer science from the University of Tsukuba, Tsukuba, Japan, in 2005 and 2008, respectively.

He is currently an Associate Professor with Kanazawa University, Kanazawa, Japan. Before this, he was a Senior Researcher with the National Institute of Information and Communications Technology, Tokyo, Japan. His research interests include future networks, Big Data, intelligent internet edge, Internet of Things, network security, information-centric network, artificial intelligence, quantum Internet, cyberphysical system, and wireless networks.

Dr. Li is the Secretary of the IEEE ComSoc Internet Technical Committee (ITC), and the Founder and Chair of the IEEE SIG on Big Data Intelligent Networking and the IEEE SIG on Intelligent Internet Edge. He is an Associate Editor for IEEE INTERNET OF THINGS JOURNAL, and was the Guest Editor of a set of prestigious magazines, transactions, and journals, such as *IEEE Communications Magazine*, IEEE NETWORK, and IEEE TRANSACTIONS ON NETWORK SCIENCE AND ENGINEERING. He also chaired several conferences and workshops, such as the General Co-Chair of the IEEE MSN 2021, the AIVR2019, and the IEEE INFOCOM 2019/2020/2021 ICCN workshop, and the TPC co-Chair of the IWQoS 2021, the IEEE MSN 2020, the BRAINS 2020, the IEEE ICDCS 2019/2020 NMIC workshop, and the ICCSSE 2019. He is a member of IEICE.

Pengfei Jiao (Member, IEEE) received the Ph.D. degree in computer science from Tianjin University, Tianjin, China, in 2018.

From 2018 to 2021, he was a Lecturer with the Center of Biosafety Research and Strategy, Tianjin University. He is currently a Professor with the School of Cyberspace, Hangzhou Dianzi University, Hangzhou, China. His research interests include complex network analysis and its applications.