Online Distributed Offloading and Computing Resource Management With Energy Harvesting for Heterogeneous MEC-Enabled IoT

Shichao Xia, Zhixiu Yao, Yun Li, *Member, IEEE*, and Shiwen Mao, *Fellow, IEEE*

***Abstract*— With the rapid development and convergence of the mobile Internet and the Internet of Things (IoT), computing-intensive and delay-sensitive IoT applications (APPs) are proliferating with an unprecedented speed in recent years. Mobile edge computing (MEC) and energy harvesting (EH) technologies can significantly improve the user experience by offloading computation tasks to edge-cloud servers as well as achieving green and durable operation. Traditional centralized strategies require precise information of system states, which may not be feasible in the era of big data and artificial intelligence. To this end, how to allocate limited edge-cloud computing resource on demand, and how to develop heterogeneous task offloading strategies with EH in a more flexible manner are remaining challenges. In this paper, we investigate an EH-enabled MEC offloading system, and propose an online distributed optimization algorithm based on game theory and perturbed Lyapunov optimization theory. The proposed algorithm works online and jointly determines heterogeneous task offloading, on-demand computing resource allocation, and battery energy management. Furthermore, to reduce the unnecessary communication overhead and improve the processing efficiency, an offloading pre-screening criterion is designed by balancing battery energy level, latency, and revenue. Extensive simulations are carried out to validate the effectiveness and rationality of the proposed approach.**

***Index Terms*— Internet of Things, mobile edge computing, energy harvesting, game theory, perturbed Lyapunov optimization.**

I. Introduction

DRIVEN by the rapid development of the Internet of Things (IoT) and the popularization of intelligent terminals (e.g., smartphones, wearable devices, autonomous driving vehicles, and intelligent sensors), cloud-oriented applications (e.g., virtual reality (VR), augmented reality (AR), autonomous driving, and online gaming) which are computation-intensive and delay-sensitive, are proliferating at an unprecedented scale in recent years [1]–[3]. Although the processing capability and the storage capacity of mobile devices (MDs) have been constantly improved, the computational performance and battery life remain serious challenges in the age of big data and artificial intelligence. As an emerging computing paradigm, mobile edge computing (MEC) significantly enhances users' service experience since some or all of the computation tasks can be offloaded to edge-cloud servers. In MEC, the computing and storage resources are deployed at the network edge to effectively reduce computation latency and avoid congestion [4].

Limited by the size and cost of hardware, the battery capacity is finite, which cannot satisfy the long-term endurance requirements of the MDs. In particular, it may even be impossible or extremely expensive to install rechargeable batteries or supply power by the traditional grid (e.g., when MDs are distributed in remote or hazardous environments). Therefore adopting cheaper, more convenient and reliable energy supply modules is increasingly essential. Fortunately, energy harvesting (EH) technologies allow MDs to capture renewable energy from the environment (e.g., solar radiation, wind, and mechanical energy) for data communications and task processing, and become increasing important to achieve green communications and durable operation of the MDs. Integrating EH into the MEC system has realistic significance [5].

The convergence of EH and MEC brings about new challenges to ensure the stability of system performance in the long-term evolution. A few recent works have considered the integration of EH and MEC and optimized tasked offloading [6]–[8]. For example, Mao *et al*. in [6] investigated an EH-enabled MEC system and proposed a low-complexity centralized computation task offloading algorithm based on Lyapunov optimization under a point-to-point communication scenario with only a single MD and a single MEC server. The authors in [7] studied the trade-off between energy consumption and stability of the battery level for the point-to-point communication scenario. Zhang *et al*. in [8] jointly optimized energy harvesting and transmit power control under the delay constraint for EH-enabled mobile cloud computing. These prior works are based on the original and simplified centralized network architecture that considers the average rate, delay, connection density, and differentiated services. However, with the rapid growth of edge devices and data in the age of IoT, centralized optimization could not be effective for

Manuscript received January 25, 2021; revised April 24, 2021; accepted April 25, 2021. Date of publication May 5, 2021; date of current version October 11, 2021. This work was supported in part by the National Natural Science Foundation of China under Grant 62071077 and Grant 61671096 and in part by the NSF under Grant ECCS-1923717. This article was presented in part at the IEEE ICC 2020. The associate editor coordinating the review of this article and approving it for publication was D. Niyato. *(Corresponding author: Yun Li.)*

Shichao Xia, Zhixiu Yao, and Yun Li are with the Chongqing Key Laboratory of Mobile Communications Technology, Chongqing University of Posts and Telecommunications, Chongqing 400065, China (e-mail: xiashichao65@163.com; zhixiuyao@163.com; liyun@cqupt.edu.cn).

Shiwen Mao is with the Department of Electrical and Computer Engineering, Auburn University, Auburn, AL 36849 USA (e-mail: smao@ieee.org).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TWC.2021.3076201.

Digital Object Identifier 10.1109/TWC.2021.3076201

distributed MEC scenarios with thousands of heterogeneous IoT applications.

In this paper, we consider a typical EH-enabled distributed MEC system with multiple edge-cloud servers and EH-enabled devices, in which the EH-enabled devices harvest energy from the surrounding environment, and offload computation tasks, when needed, to available edge-cloud servers. It should be noted that the applications running in the EH-enabled devices have heterogeneous data characteristics and requirements. For example, wearable devices or smart home applications have low data rate, high latency tolerance, and plain battery energy level demand. In the case of autonomous vehicles with rigorous battery energy level management, each vehicle could generate at least 100GB of data per day, which are also extremely sensitive to latency [9]. Meanwhile, the edge-cloud servers have heterogeneous edge computing resources (e.g., different service providers have different computation and storage performance, and ask different service prices). Moreover, driven by the MEC and EH technologies, more applications will offload tasks to the cloud for improved computational efficiency, which poses serious challenges to enhance the efficiency and fairness of edge-cloud resource allocation.

Inspired by these observations, we aim to address the following main issues in this paper: (i) how to make the optimal offloading and energy harvesting decisions to ensure stability of the battery energy level and guarantee the computation performance in a distributed manner; (ii) how to allocate edge-cloud computing resources, provide on-demand services, and ensure fairness for heterogeneous IoT applications; and (iii) how to maximize the time-averaged network utility, and balance the system energy consumption and computation latency.

The major contributions of this work are summarized as follows.

- We investigate the joint problem of heterogeneous task offloading scheduling, computing resource allocation, and EH management. Task offloading scheduling and computing resource allocation are formulated as a distributed optimization problem, which also aims to stabilize the battery energy level and guarantee the computation performance in the long-term evolution.
- To determine the optimal strategies of heterogeneous task offloading and computing resource on-demand allocation, and to ensure fairness, a dynamic quote price mechanism for edge-cloud resource is designed based on the game-theoretic approach. Moreover, the optimal strategies are proven to be the *Stackelberg Equilibrium* (SE) solution.
- The offloading and pricing decisions rely on multiple pieces of time-dependent information (e.g., battery level, computation task backlog, computing resource, and harvested energy). To maximize the time-averaged network utility and ensure system computation performance, we decouple task offloading, energy harvesting, and computing resource allocation with the perturbed Lyapunov optimization theory and develop effective online, distributed algorithms.
- To reduce the unnecessary communication signaling overhead and select suitable edge-cloud servers, an offloading pre-screening criterion is proposed to balance battery energy level, computation latency, and revenue.

The remainder of this paper is organized as follows. Section II reviews main related works. In Section III, we describe the system model and formulate the computation task offloading and energy harvesting problem. Online distributed computation offloading and resource management with Lyapunov optimization and game theory is presented in Section IV. Section V analyzes the game strategies and proves the existence of the SE. Section VI designs an offloading pre-screening criterion and a price update function. Section VII evaluates the performance of the proposed scheme. Section VIII concludes this paper.

II. Related Work

Due to the limitation of processing capability, storage capacity, and battery life of traditional smart mobile devices (MDs), how to further improve the computational performance and battery endurance of MDs has become one of the major questions in the new era of network reform [5]. Computation offloading and energy harvesting (EH) have been recognized as important technologies to enhance mobile user experience as well as achieving green and durable communications, and have attracted much attention in recent years [10].

Considerable effort has been made on how to effectively offload tasks to center/edge-cloud servers. To minimize energy consumption under delay constraint, Deng *et al*. in [11] formulated a task allocation problem based on the interplay and cooperation of fog and cloud, aiming to balance power consumption and transmission delay to optimize workload allocations toward fog and cloud. Dai *et al*. in [12] addressed the computation offloading problem with particle swarm optimization in the Internet of Vehicles and multi-access edge computing environment, in which health applications were divided into different parts to offload to nearby vehicles. To maximize the energy efficiency, Liu *et al*. in [13] formulated a joint radio and computational resource allocation problem by applying the non-orthogonal multiple access (NOMA) technique to improve the energy efficiency of NOMA-enabled MEC. Moreover, energy efficiency optimization is important for resource-constrained e-Health devices. You *et al*. in [14] proposed a prior threshold-based task offloading method, and formulated optimal resource allocation as a convex optimization problem to maximize energy efficiency in a centralized manner.

Energy harvesting helps to make MDs self-sustainable and to achieve the goal of green communications [15]–[19]. Considering the random arrival of sensory data and harvested energy, Yang *et al*. in [15] minimized transmission delay by optimizing the data rate. Gupta *et al*. in [16] analyzed two opportunistic transmission mechanisms in energy harvesting wireless sensor networks, and studied the performance of sequential transmissions. Under the constraint of harvested energy, Li *et al*. in [17] investigated the transmit power control strategy in mobile sensor networks to minimize the system energy consumption. Calvo-Fullana *et al*. in [18] considered how to select the number of energy harvesting sensors in an energy harvesting enabled sensor network, and

studied transmit power control to avoid data fusion distortion. Guo *et al*. in [19] discussed how to optimize the time ratio of energy transmission to maximize the network energy efficiency of wireless data and energy transmission. Considering two-way relay cooperation network composed of two source nodes and an energy harvesting node, Hu *et al*. in [20] proposed an optimization algorithm to minimize the number of long-term average power outage under the constraint of long-term average battery based on Lyapunov optimization theory.

In recent years, the convergence of MEC and EH technologies has been seen as an advanced and promising architecture to provide efficient, convenient, and flexible services [21]–[25]. Wang *et al*. in [21] jointly optimized wireless access mode, computing and wireless resource allocation to minimize the energy consumption in heterogeneous networks. Considering the dynamics and randomness of wireless channels and task arrival process, Teng *et al*. in [22] proposed a hybrid time-scale task offloading and wireless resource allocation strategy to minimize system energy consumption. To minimize the total computing and transmission energy consumption, Zhang *et al*. in [23] jointly optimized the computing resources, wireless bandwidth, and transmit power with Lyapunov optimization theory. Zhuang *et al*. in [24] modeled computational resource and transmission power allocation as a convex optimization problem to minimize the end-to-end task processing delay. Mahmood *et al*. in [25] jointly optimized the harvested energy, computing resource of devices, and wireless bandwidth resources to maximize the computing energy efficiency of the system. Moreover, without requiring a priori knowledge of the network, Chen *et al*. in [26] proposed a dynamic task offloading algorithm based on a double deep Q-network to maximize the long-term revenue utility.

III. System Model and Problem Formulation

In the energy harvesting (EH)-enabled MEC offloading system, it is essential for each mobile device (MD) to determine the local computing and task offloading strategies on the basis of the devices' status (such as battery energy level, task type, and backlog). To maximize the utilization of edge-cloud resources, the edge-cloud servers should allocate computing resources on demand, e.g., by allocating virtual machines with different CPU frequencies to different requests [27]. In this section, we first model the EH-enabled MEC offloading network, where each MD is capable of EH. Next, the offloading and computing resource allocation problem is formulated.

We consider an EH-enabled MEC network system as shown in Fig. 1. The system serves a set of MDs, $\mathcal{M} = \{1, 2, \ldots, m\}$, each equipped with an energy-harvesting component (e.g., piezoelectric transducer [28] and electromagnetic transducer [29]) and fully powered by harvested renewable energy. There is a set of edge-cloud servers, $\mathcal{N} = \{1, 2, \ldots, n\}$, which provide computing or data storage services to the MDs within its radio coverage. The MDs can process tasks locally or offload tasks to the edge-cloud servers. We assume that the system operates in discrete time slots and let τ and $\mathcal{T} \triangleq \{0, 1, \ldots, \}$ denote the duration of each time slot and the time slot indices, respectively.

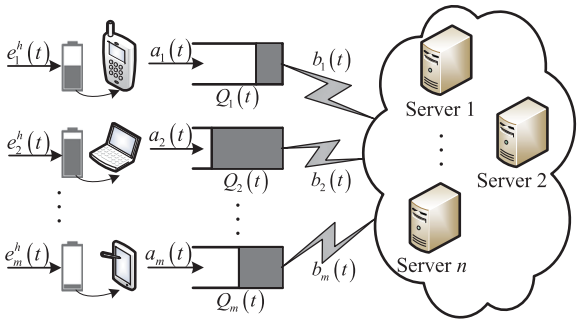


Fig. 1. Energy harvesting enabled MEC offloading system: (i) Each MD is capable of energy harvesting; (ii) Each MD can locally execute tasks or offload them to the edge-cloud servers.

A. Computation Task and Queueing Model

The computation task of MDs follows the data-partition model [9], [30], namely, the task-input bits are bit-wise independent and can be arbitrarily divided into different groups, and executed at the MDs or the edge-cloud servers. Different MDs with diverse requirements (such as offloading deadline and energy constraint) have different task characteristics (called the task's "DNA"), e.g., data type, task size, and computation density. In time slot t, the task's "DNA" of MD i can be characterized by a three-tuple $\Lambda_i(t) = \langle b_i(t), \tau_i^d, \gamma_i \rangle$[1], where $b_i(t)$ is the processed task (include programs codes, configurations information, etc.) in time slot t, τ_i^d is the maximum computation latency of $b_i(t)$, and γ_i is the computation density (in cycles/bit), which can be obtained through off-line measurement [31].

The task arrivals of the MDs are modeled as an independent, identically distributed (i.i.d.) Bernoulli process. Let $a_i(t)$ and $Q_i(t)$ denote the task arrival and task queue backlog at MD i in time slot t, respectively. Accordingly, let $\mathcal{A}(t) = \{a_1(t), a_2(t), \ldots, a_m(t)\}$ and $\mathcal{Q}(t) = \{Q_1(t), Q_2(t), \ldots, Q_m(t)\}$ denote the arrived task set and queue backlog set of all MDs in time slot t, respectively. As the arrived task in each time slot is finite, we have $0 \leq a_i(t) \leq a_i^{\max}, i \in \mathcal{M}, t \in \mathcal{T}$, and $\mathbb{E}\{\mathcal{A}(t)\} = \{\lambda_1, \ldots, \lambda_m\}$, where λ_i is the task arrival rate of MD i. The task backlog at MD i evolves as follows.

$$Q_i(t+1) = \max\{Q_i(t) - b_i(t), 0\} + a_i(t), \quad i \in \mathcal{M}, t \in \mathcal{T}. \quad (1)$$

B. Local Processing Model

At the beginning of each time slot, the MDs should decide whether to offload or not, how much task will be offloaded, and where to offload. For local processing, the MD needs to allocate computing resource (e.g., computing power or CPU frequency) to process the task.

1) Computation Task Model: To achieve energy savings under latency constraints, the MDs should process tasks at an appropriate CPU clock frequency determined by the dynamic voltage frequency scaling (DVFS) technique [32]. Let $b_{i0}(t)$

[1]To simplify analysis, we assume that the computing deadline and density of the processed task do not change in a time slot.

denote the locally executed task of MD i in time slot t, given by

$$b_{i0}(t) = \int_{\tau} f_{i0}(t)/\gamma_i \, dt, \quad i \in \mathcal{M}, t \in \mathcal{T}, \tag{2}$$

where $f_{i0}(t)$ is the CPU clock speed of MD i.[2] Considering that the CPU clock is bounded, we have $f_{i0}^{\min} \leq f_{i0}(t) \leq f_{i0}^{\max}$.

2) Local Computing Energy Consumption Model: Considering the limitation of battery energy, the local execution and offloading decisions should take energy consumption into account. For each MD, energy consumption occurs when processing local task (computing energy consumption) and offloading a task (communication energy consumption).[3]

When a task is processed locally, we assume that the main energy consumption is due to the CPU, and ignore other types of energy consumption at the MD. Typically, the CPU energy consumption to process task $b_{i0}(t)$ is proportional to the CPU frequency, modeled as follows.

$$e_{i0}^{p}\{f_{i0}(t)\} = \kappa_i \int_{\tau} (\alpha(f_{i0}(t))^{\sigma} + \beta)\, dt, \tag{3}$$

where κ_i is the effective energy coefficient associated with the chip architecture; α and β are the parameters determined by the CPU model; and σ ranges from 2 to 3, which can be obtained through off-line measurements [33]. We set $\alpha = 1$, $\beta = 0$, and $\sigma = 2$ in this paper.

C. Edge-Cloud Processing Model

Edge-cloud servers have sufficient power supply, and more computation and storage capabilities compared with MDs. Once the MD decides to offload a task, the offloaded task will be transmitted to a server through a wireless channel, and then the server will allocate appropriate virtual machine resource (mainly including computing and storage resources) for the MD. The communication model (i.e., communication latency and energy consumption) of MDs and the edge-cloud processing model will be presented in the following.

1) Communication Model: Let $h_i(t) = [l(t)]^o$ denotes the wireless channel power gain, where $l(t)$ is the communication distance in time slot t, and $o \in \{2, 3\}$ is a constant. According to the Shannon-Hartley theory, the task transmission rate of MD i in time slot t is given by $r_i[P_i(t), h_i(t)] = B_i \log_2\left(1 + \frac{P_i(t)h_i(t)}{w}\right)$, in which B_i, $P_i(t)$ and w are the channel bandwidth, transmit power, and average noise power in time slot t, respectively.

For MD i, let $I_{ij}(t) \in \{0, 1\}$, $j \in \mathcal{N}$, $t \in \mathcal{T}$ be the computation task offloading strategy indicators, where $I_{ij}(t) = 1$ denotes that MD i will offload task to the jth edge-cloud server in time slot t. Thus, the transmission delay of MD i can be derived as follows.

$$d_{ij}^{c}(t) = \frac{b_{ij}(t) \cdot \mathbf{1}\{I_{ij}(t) = 1\}}{r_i[P_i(t), h_i(t)]}, \quad j \in \mathcal{N}, t \in \mathcal{T}, \tag{4}$$

where $\mathbf{1}\{\cdot\}$ is the indicator function. Accordingly, we can obtain the communication energy consumption of MD i for offloading task $b_{ij}(t)$ to edge-cloud server j as follows.

$$e_{ij}^{c}\{b_{ij}(t), I_{ij}(t)\} = P_i(t)\, d_{ij}^{c}(t). \tag{5}$$

Next, we present the communication cost model. Let $\varphi_{ij}(t)$ denote the unit uplink communication cost from MD i to edge-cloud server j in time slot t.[4] The communication cost can be expressed as follows.

$$c_{ij}(t) = \varphi_{ij}(t)\, b_{ij}(t) \cdot \mathbf{1}\{I_{ij}(t) = 1\}, \quad i \in \mathcal{M}, j \in \mathcal{N}. \tag{6}$$

2) Edge-Cloud Processing Delay Model: Let $f_{ij}(t)$ denote the CPU clock frequency (in $cycle/s$) of the allocated virtual machines at edge-cloud server j for MD i in time slot t. Since the CPU clock frequency is constrained by the maximum $f_j^{\min}$ and minimum $f_j^{\max}$, we have $f_j^{\min} \leq f_{ij}(t) \leq f_j^{\max}$. Accordingly, the processing delay of edge-cloud server j is

$$d_{ij}^{p}(t) = \frac{b_{ij}(t) \cdot \mathbf{1}\{I_{ij}(t) = 1\}}{\gamma_i f_{ij}(t)}, \quad j \in \mathcal{N}. \tag{7}$$

3) Edge-Cloud Execution Energy Consumption Model: The computation energy consumption of edge-cloud server j to process task $b_{ij}(t)$ can be calculated as follows.

$$e_{ij}^{p}(t) = \kappa_j \int_{\Delta t} (f_{ij}(t))^2 dt, \quad j \in \mathcal{N}, \tag{8}$$

where $\Delta t = d_{ij}^{p}(t)$ is the processing delay of edge-cloud server j.

D. Energy Harvesting Model

Assume that the harvested energy arrival process of MDs is i.i.d. in different time slots. Let $\delta_i(t)$ denote the harvested energy in time slot t for MD i with the maximum value $\delta_i^{\max} = P_i^h \tau$, and P_i^h be the average energy harvesting power in the system, which can be obtained through off-line measurements [34].[5] Considering that only part of the harvested energy can be stored in the battery in practice, let $e_i^h(t)$ represent the harvested energy charged to battery of MD i in time slot t. We have

$$0 \leq e_i^h(t) \leq \delta_i(t), \quad i \in \mathcal{M}, \ t \in \mathcal{T}. \tag{9}$$

From Section III-C, we obtain the total energy consumption of MD i in time slot t as follows.

$$e_{i0}^{t}(t) = e_{i0}^{p}(t) + \sum_{j=1}^{n} e_{ij}^{c}(t) \cdot \mathbf{1}\{I_{ij}(t) = 1\}. \tag{10}$$

To prevent over-discharging of battery [36], we define the battery discharging constraint as

$$E_i^{\min} \leq e_{i0}^{t}(t) \leq E_i^{\max}, \tag{11}$$

[2]Note that, it has been proved that the optimal computing frequency should be the same within each time slot [6]. Hence the computation task can be rewritten as $b_{i[\cdot]}(t) = f_{i[\cdot]}(t)\tau/\gamma_i$.

[3]In this paper, we assume that the energy harvested is always used to process local tasks and offloading.

[4]Note that, the communication cost of each MD is related not only to the size of the task, but also to the communication mode. For example, the cost of using cellular communication is higher than that using WiFi. Although the model adopted is simple, it captures the nature of communication cost and can be easily extended to other communication models.

[5]Generally, energy harvesting power P_i^h is related to the energy-harvesting components, e.g., a wind turbine generates around 100mW at wind speeds 2m/s $\sim$ 9m/s, and dedicated radio frequency (RF) sensors can produce 5μW at a transmission power of 4W and a distance of 15m [35].

where $E_i^{\min}$ and $E_i^{\max}$ are the minimum and maximum discharged energy of the battery in each time slot, respectively.

In order to ensure continuous operation of MDs, the battery energy level must be sufficient to power for local task processing and communications at the beginning of each time slot t. Let $B_i(t)$ denote the battery energy level of MD i at the beginning of time slot t. Therefore, we have the constraint of energy consumption of MD i in time slot t as follows.

$$E_i^{\min} \le e_{i0}^t(t) \le \min\left\{E_i^{\max}, B_i(t)\right\} < \infty. \tag{12}$$

If the inequalities in (12) does not hold, the task has to be stored in the backlog queue. According Eqs. (9) and (10), we obtain the dynamics of the energy queue of MD i as follows.

$$B_i(t+1) = \max\left\{B_i(t) - e_{i0}^t(t), 0\right\} + e_i^h(t). \tag{13}$$

E. Task Offloading Utility Model

In order to evaluate the benefit of processing task in time slot t, we adopt the logarithmic utility function for each MD i, which has been widely used in the wireless communications and mobile computing domain as follows [37]–[39].

$$u_{i\hat{j}}(t) = \rho_i \log\left(1 + b_{i\hat{j}}(t)\right),\ i \in \mathcal{M}, \hat{j} \in \{0, \mathcal{N}\}, \tag{14}$$

where ρ_i is a benefit weight parameter of MD i.

F. Revenue Maximization Problem

The EH-enabled MEC offloading system aims to ensure that each MD has sufficient energy to execute the offloading strategy, and to meet the task queue stability and task offloading deadline constraints of each MD in each time slot. Correspondingly, we define the offloading decisions, processed task, computing resource allocation profiles, and energy harvesting decisions of the MEC offloading system in time slot t as $\boldsymbol{I}(t) = \{I_{ij}(t)\}_{i\in\mathcal{M}, j\in\mathcal{N}}$, $\boldsymbol{b}(t) = \left\{b_{i\hat{j}}(t)\right\}_{i\in\mathcal{M}, \hat{j}\in\{0,\mathcal{N}\}}$, $\boldsymbol{F}(t) = \left\{f_{i\hat{j}}(t)\right\}_{i\in\mathcal{M}, \hat{j}\in\{0,\mathcal{N}\}}$, and $\boldsymbol{e}(t) = \left\{e_i^h(t)\right\}_{i\in\mathcal{M}}$, respectively. According to the task offloading decision, resource allocation, and energy consumption in each time slot, we define the revenue maximization problem as follows.

$$\mathcal{P}1: \max_{\boldsymbol{I}(t),\boldsymbol{b}(t),\boldsymbol{F}(t),\boldsymbol{e}(t)} R = \lim_{T\to\infty}\frac{1}{T}\mathbb{E}\times\left[\sum_{t=0}^{T-1}\left\{\sum_{\forall i,\hat{j}}\left(u_{i\hat{j}}(t) - c_{ij}(t) - \psi_j e_{ij}^p(t)\right)\right\}\right] \tag{15}$$

$$s.t.\ (9), (12)$$

$$0 \le \sum_{\forall \hat{j}} b_{i\hat{j}}(t) \le Q_i(t),$$

$$\forall i \in \mathcal{M},\quad \hat{j} \in \{0, \mathcal{N}\},\ t \in \mathcal{T} \tag{16}$$

$$f_{\hat{j}}^{\min} \le f_{i\hat{j}}(t) \le f_{\hat{j}}^{\max},$$

$$\forall i \in \mathcal{M},\quad \hat{j} \in \{0, \mathcal{N}\},\ t \in \mathcal{T} \tag{17}$$

$$\bar{Q}_i = \lim_{T\to\infty}\sup\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left\{Q_i(t)\right\} < \infty, t \in \mathcal{T}, \tag{18}$$

where ψ_j is the unit energy consumption cost of edge-cloud server j. Inequality (16) ensures that the sum of the processed tasks for local execution and offloading does not exceed the queue backlog of MD i in time slot t; inequality (17) indicates that the allocated CPU frequency is within the maximum and minimum range in each time slot; and inequality (18) is the stability constraint of task queues.

IV. Online Distributed Dynamic Computation Task Offloading Algorithm

In the age of IoT, both edge devices and data volume are growing rapidly. On one hand, it is difficult or even impossible to collect the global system status information in real-time. And the traditional centralized optimization methodologies may not be applicable to the distributed MEC scenarios with thousands of heterogeneous IoT applications. On the other hand, due to the intermittent, heterogeneous, and sporadic natures of the task arrivals and harvested energy, accurate prediction of system status is also impossible. As a result, we develop a distributed dynamic computation task offloading and computing resource allocation strategy based on the buyer-seller game theory and perturbed Lyapunov optimization in this section. We will transform the centralized optimization problem $\mathcal{P}1$ to a distributed optimization problem $\mathcal{P}2$.

A. Optimization Model Based on Buyer/Seller Game Theory

Processing the offloaded tasks at the edge-cloud servers incur costs (e.g., computing energy consumption, hardware cost, etc.), which should be covered by the MDs [42]. This can be viewed as a "market" in which each MD buys task processing products from suitable edge-cloud servers. The MDs can be regarded as buyers (denoted by $\boldsymbol{b}$) who buy virtual machines (or, computing resource) to process their offloaded task, and edge-cloud servers can be viewed as sellers (denoted by $\boldsymbol{s}$) who provide computing services to the buyers.

The payment from the buyers (i.e., MDs) should be positively related to their task offloaded to the sellers (i.e., edge-cloud servers). In this section, we first define the unit task payment of MD i who offload a task to the jth edge-cloud server as $p_{ij}(t)$ (in $\$/bit$) in time slot t. Hence, we can define the offloading payment $s_{ij}(t)$ as follows.

$$s_{ij}(t) = p_{ij}(t)\, b_{ij}(t), i \in \mathcal{M}, j \in \mathcal{N}. \tag{19}$$

1) Buyers/MDs Game model: We assume that the MDs are always rational and seek to maximize their individual revenue. It is obviously that the optimal strategy of buyers should take offloaded task, communication costs, and payments into account. According to (6), (14) and (19), the object function of buyer i in time slot t is given by

$$u_{b_i}(t) = \sum_{\forall \hat{j}}\left\{u_{i\hat{j}}(t) - c_{ij}(t) - s_{ij}(t)\right\},$$

$$\hat{j} \in \{0, \mathcal{N}\},\quad t \in \mathcal{T}, \tag{20}$$

To maintain the stability of the battery energy level and guarantee the computation performance in the long-term evolution, we define problem $\mathcal{P}2-buyer$ for buyer i as follows.

$$\mathcal{P}2-buyer: \max_{\boldsymbol{I}_i(t),\boldsymbol{b}_i(t),\boldsymbol{e}_i(t)} u_{b_i}$$
$$= \lim_{T\to+\infty}\frac{1}{T}\mathbb{E}\left[\sum_{t=0}^{T-1}\left\{\sum_{\forall\hat{j}}\left(u_{i\hat{j}}(t)-c_{ij}(t)-s_{ij}(t)\right)\right\}\right]$$
$$\text{s.t. } (9),\ (12),\ (16)--(18). \tag{21}$$

2) Sellers/edge-Cloud Servers Game Model: Let s_{ji} denote the reward that edge-cloud server j receives by providing computing resource for MD i. According to (19), the revenue of the jth edge-cloud server $s_{ji}(t)$ in time slot t is

$$s_{ji}(t)=s_{ij}(t)=p_{ij}(t)\,b_{ij}(t),\ i\in\mathcal{M},j\in\mathcal{N}. \tag{22}$$

According to (8) and (22), we define the optimization problem of seller j as follows.

$$\mathcal{P}2-seller: \max_{\boldsymbol{I}_j,\boldsymbol{P}_{ij},\boldsymbol{F}_{ij}} u_{s_j}(t) \tag{23}$$
$$= \lim_{T\to\infty}\frac{1}{T}\mathbb{E}\left[\sum_{t=0}^{T-1}\left\{\sum_{i=1}^{m}\left(s_{ji}(t)-\psi_j e_{ij}^{p}(t)\right)\right\}\right] \tag{24}$$
$$\text{s.t. } p_{ij}(t)\geq 0, i\in\mathcal{M},t\in\mathcal{T}. \tag{25}$$

B. Game Model Problem Analysis

According to (1) and (13), the optimal strategies of both the buyers and the sellers are only related to their current states, and are irrelevant to the states in the past. Therefore, problems $\mathcal{P}2-buyer$ and $\mathcal{P}2-seller$ are markov decision process (MDP) problem, and can be solved by MDP or reinforcement learning (RL) approaches [26]. Unfortunately, compared to the conventional MEC system with battery-power devices, the computation offloading and pricing decisions designed for the EH-enable MEC system are much more complicated because the CPU clock speed ($f_{i\hat{j}}(t)$), task request ($a_i(t)$), harvestable energy units ($\delta_i^{\max}$), wireless channel state ($h_i(t)/w$), battery energy levels ($B_i(t)$), task cache backlogs ($Q_i(t)$), and price ($p_{ij}(t)$) need to be considered in each time slot. To use decentralized MDP or RL to solve problem $\mathcal{P}2-buyer$, we have to consider $|f_{i0}(t)|\cdot|a_i(t)|\cdot|\delta_i^{\max}|\cdot|h_i(t)/w|\cdot|B_i(t)|\cdot|Q_i(t)|\cdot|p_{ij}(t)|$ states, where $|x|$ denote the number of states of x. Assuming the average number of states for each item is 100, the total number of states will reach 10^{14}, which requires a lot of computing power and huge storage space. This is almost impossible to implement in practice for each MD with limited computing and storage capabilities.

Therefore, we design an online task offloading and energy management scheme based on game theory with perturbed Lyapunov optimization, with greatly reduced complexity and storage requirements.

C. Game Model Analysis Based on Perturbed Lyapunov Optimization

1) Buyers/MDs Game Model Based on Perturbed Lyapunov Optimization: It should be noted that due to the battery energy causality constraints (11) and (13), the battery energy level is time-dependent. We first define two important parameters: the perturbation parameter θ_i and the virtual energy queue $\tilde{B}_i(t)$ of the battery at MD i [41] in the following.

Definition 1: The perturbation parameter θ_i is a bounded constant given by

$$\theta_i\geq\tilde{E}_i^{\max}+V_i\left(E_i^{\min}\right)^{-1}, \tag{26}$$

where $\tilde{E}_i^{\max}=\min\left\{\kappa_i(f_{i0}^{\max})^2\tau_i^d+\sum_{j=1}^{n}P_i(t)\,\tau_i^d,E_i^{\max}\right\}$.

Definition 2: Let $\tilde{B}_i(t)=B_i(t)-\theta_i$ denote the virtual energy queue to track the battery energy level at MD i. By carefully setting θ_i, there will be enough battery power to maintain the local task execution and communications for MD i.

Theorem 1: The battery energy level always satisfies $0\leq B_i(t)\leq\theta_i+\delta_i^{\max}$ in each time slot.

Proof: Please refer to Appendix A. □

Then, we define the Lyapunov function for the computation task queue and battery virtual energy queue as follows.

$$L[\Theta_i(t)]=\frac{1}{2}\left\{(Q_i(t))^2+\left(\tilde{B}_i(t)\right)^2\right\}. \tag{27}$$

Obviously, $L[\Theta_i(t)]\geq 0$. According to [40], the Lyapunov Drift is given by

$$\Delta[\Theta_i(t)]=\mathbb{E}\left\{L[\Theta_i(t+1)]-L[\Theta_i(t)]\,|Q_i(t),\tilde{B}_i(t)\right\} \tag{28}$$

The underlying objective of the online optimal decision is to minimize the upper bound of the *drift-minus-utility* function, which is defined as follows.

$$\Delta[\Theta_i(t)]-V_i\mathbb{E}\left\{u_{b_i}(t)\,|\Theta_i(t)\right\}, \tag{29}$$

where $V_i\geq 0$ is a non-negative controllable parameter. We have ***Theorem 2*** for an upper bound of the drift-minus-utility.

Theorem 2: For any given control parameter $V_i\geq 0$, $a_i(t)\in[0,a_i^{\max}(t)]$, and $e_i^h(t)\in[0,\delta_i^{\max}]$, we have

$$\begin{aligned}&\Delta[\Theta_i(t)]-V_i\mathbb{E}\left\{u_{b_i}(t)\,|\Theta_i(t)\right\}\\ &\leq\ \mathbb{E}\left\{B_i(t)\left[e_i^h(t)-e_{i0}^t(t)\right]|\Theta_i(t)\right\}\\ &\quad+\mathbb{E}\left\{Q_i(t)\left[a_i(t)-b_{ij}(t)\right]|\Theta_i(t)\right\}\\ &\quad+\Phi_i-V_i\mathbb{E}\left\{u_{b_i}(t)\,|\Theta_i(t)\right\},\end{aligned} \tag{30}$$

where Φ_i is a nonnegative constant given by $\Phi_i=\frac{1}{2}\left\{(E_i^{\max})^2+(\delta_i^{\max})^2\right\}+\frac{1}{2}\left\{(b_i^{\max})^2+(a_i^{\max})^2\right\}$.

Proof: Please refer to Appendix B. □

It can be seen that minimizing the upper bound of drift-minus-utility is equivalent to minimize the Right-Hand-Side (RHS) of inequality (30). According to *Theorem 2*, we can transform the optimization problem $\mathcal{P}2-buyer$ to $\mathcal{P}2-buyer'$ as follows.

$$\mathcal{P}2-buyer': \max_{\boldsymbol{I}_i(t),\boldsymbol{b}_i(t),\boldsymbol{e}_i(t)} U_{b_i}(t)$$
$$=V_iu_{b_i}(t)+Q_i(t)\left[b_i(t)-a_i(t)\right]$$
$$+\tilde{B}_i(t)\left[e_{i0}^t(t)-e_i^h(t)\right]$$
$$\text{s.t. } (9),(11),(16)-(18). \tag{31}$$

2) Sellers/Edge-Cloud Game Model: Furthermore, we can transform the optimization problem $\mathcal{P}2-seller$ to $\mathcal{P}2-seller'$ based on the maximum value theory as follows.

$$\begin{aligned}\mathcal{P}2-seller': &\max_{\boldsymbol{I}_j,\boldsymbol{P}_{ij},\boldsymbol{F}_{ij}} U_{s_j}(t)\\ &=\sum_{\forall i}\left(s_{ji}(t)-\psi_j e^p_{ij}(t)\right), i\in\mathcal{M}, t\in\mathcal{T}\\ &\text{s.t. } (25). \end{aligned} \tag{32}$$

V. Optimal Game Strategies and Stackelberg Equilibrium

In this section, we first analyze the optimal energy harvesting, task offloading, and computing resource allocation strategies. We will then show that the optimal solutions are *Stackelberg Equilibrium* (SE) solutions.

A. Optimal Strategy for Buyers/MDs

For each mobile device (MD), there are three fundamental questions need to be answered: (i) how much harvested energy should be stored in the battery; (ii) which task should be processed locally; and (iii) how to select a suitable server for offloading.

1) Optimal Strategy for Energy Harvesting: According to (31), we can easily drive the optimal strategy for energy harvesting as the solution to the following problem.

$$\max_{0\le e^h_i(t)\le\delta_i(t)} -\tilde{B}_i(t)\, e^h_i(t). \tag{33}$$

Solving the problem, we obtain the optimal amount of energy to harvest in time slot t as follows.

$$\left[e^h_i(t)\right]^* = \delta_i(t)\cdot\mathbf{1}\left\{\tilde{B}_i(t)<0\right\}. \tag{34}$$

If $\tilde{B}_i(t)<0$, MD i should store the maximum amount of energy $\delta_i(t)$. Otherwise, MD i does not store any energy.

2) Optimal Offloading Strategy: According to (31) and (33), we obtain the new optimization problem of buyers as follows.

$$\begin{aligned}\mathcal{P}2-buyer'': &\max_{\boldsymbol{I}_i(t),\boldsymbol{b}_i(t)} U_{b_i}(t)\\ &=V_i u_{b_i}(t)+Q_i(t)\left[b_i(t)-a_i(t)\right]\\ &\quad+\tilde{B}_i(t)\, e^t_{i0}(t)\\ &\text{s.t. } (11),(12),(16)-(18). \end{aligned} \tag{35}$$

a) Local execution strategy: Due to the battery discharging constraint (11), we can obtain the minimum and maximum allocated CPU frequency for MD i as $f^L_{i0}(t)=\max\left\{f^{\min}_i,\sqrt{E^{\min}_i-\sum_{j=1}^n P_i\frac{b^*_{ij}}{r_i}/\{\kappa_i\tau\}}\right\}$ and $f^U_{i0}(t)=\min\left\{f^{\max}_{i0},\sqrt{E^{\max}_i-\sum_{j=1}^n P_i\frac{b^*_{ij}}{r_i}/\{\kappa_i\tau\}}\right\}$, respectively. If and only if $f^L_{i0}(t)\le f^U_{i0}(t)$, MD i will process the task locally.

Proposition 1: According to (35), the optimal local execution strategy for MD i is given by:

$$f^*_{i0}(t)=\begin{cases}f^L_{i0}(t), \tilde{B}_i(t)<0, f^M_{i0}(t)<f^L_{i0}(t)\\ f^M_{i0}(t), f^L_{i0}(t)<f^M_{i0}(t)<f^U_{i0}(t)\\ f^U_{i0}(t), \tilde{B}_i(t)\ge 0 \text{ or } \tilde{B}_i(t)<0, f^M_{i0}(t)\ge f^L_{i0}(t),\end{cases} \tag{36}$$

where $f^M_{i0}(t)=-\frac{A+\sqrt{A^2-8\tilde{B}_i(t)\kappa_i\tau\left(Q_i+\frac{V_i\rho_i}{\ln 2}\right)}}{4\tilde{B}_i(t)\kappa_i\tau}$, *and* $A=2\tilde{B}_i(t)\kappa_i\gamma_i+Q_i\tau/\gamma_i$.

Proof: According to (35), the first order partial derivative of $U_{b_i}(t)$ with respect to $f_{i0}(t)$ is

$$\frac{\partial U_{b_i}(t)}{\partial f_{i0}(t)}=\frac{V_i\rho_i}{\left(1+\frac{f_{i0}(t)\tau}{\gamma_i}\right)\ln 2}\frac{\tau}{\gamma_i}+Q_i\frac{\tau}{\gamma_i}+2\tilde{B}_i(t)\kappa_i f_{i0}(t)\tau. \tag{37}$$

Furthermore, the second order partial derivative of $U_{b_i}(t)$ with respect to $f_{i0}(t)$ is given by

$$\frac{\partial^2 U_{b_i}(t)}{\partial f_{i0}(t)^2}=-\frac{V_i\rho_i}{\left(1+\frac{f_{i0}(t)\tau}{\gamma_i}\right)^2\ln 2}\frac{\tau^2}{{\gamma_i}^2}+2\tilde{B}_i(t)\kappa_i\tau. \tag{38}$$

Then we have the following two cases according to (37) and (38):

- If $\tilde{B}_i(t)<0$, we have $\frac{\partial^2 U_{b_i}(t)}{\partial f_{i0}(t)^2}<0$. Moreover, the inequalities (11), (12), (16)–(18) are affine functions. Hence $U_{b_i}(t)$ is convex with respect to $f_{i0}(t)$. Defining the Lagrangian and according to the KKT (Karush-Kuhn-Tucker) conditions [43], we find the global optimum $f^M_{i0}(t)\,|_{\frac{\partial U_{b_i}(t)}{\partial f_{i0}(t)}=0}=-\frac{A+\sqrt{A^2-8\tilde{B}_i(t)\kappa_i\tau\left(Q_i+\frac{V_i\rho_i}{\ln 2}\right)}}{4\tilde{B}_i(t)\kappa_i\tau}$. Accordingly, if $f^L_{i0}(t)<f^M_{i0}(t)\le f^U_{i0}(t)$, then the optimal solution is $f^*_{i0}(t)=f^M_{i0}(t)$. If $f^M_{i0}(t)\le f^L_{i0}(t)$, then the optimal solution is $f^*_{i0}(t)=f^L_{i0}(t)$. If $f^*_{i0}(t)>f^U_{i0}(t)$, then the optimal solution is $f^*_{i0}(t)=f^U_{i0}(t)$.
- If $\tilde{B}_i(t)\ge 0$, we have $\frac{\partial U_{b_i}(t)}{\partial f_{i0}(t)}>0$. $U_{b_i}(t)$ is a monotone increasing function of $f_{i0}(t)$. Therefore, the optimal solution is $f^*_{i0}(t)=f^U_{i0}(t)$ in this case.

□

b) Offloading strategy: At the beginning of each time slot, each MD should first select one or more appropriate edge-cloud servers, which will be discussed later in Section VI-A. Once the server is chosen, we have the following proposition.

Proposition 2: For a chosen edge-cloud server j, the optimal offloaded task is given by

$$b^*_{ij}(t)=\begin{cases}b^L_{ij}(t), b^M_{ij}(t)<b^L_{ij}(t)\\ b^M_{ij}(t), b^L_{ij}(t)\le b^M_{ij}(t)<b^U_{ij}(t)\\ b^U_{ij}(t), b^M_{ij}(t)\ge b^U_{ij}(t),\end{cases} \tag{39}$$

where $b^M_{ij}=\frac{\rho_i}{C}-1$, $C=\varphi_{ij}\ln 2+p_{ij}(t)\ln 2-\frac{\ln 2}{V_i}\left(Q_i+\tilde{B}_i(t)\frac{P_i}{r_i}\right)$, *and* $b^L_{ij}(t)$ *and* $b^U_{ij}(t)$ *are the minimum and maximum offloaded task, respectively, which will be defined in Section VI-A.*

Proof: According to (35), the first order partial derivative of U_{b_i} with respect to $b_{ij}(t)$ is

$$\frac{\partial U_{b_i}(t)}{\partial b_{ij}(t)}=V_i\left\{\rho_i\frac{1}{(1+b_{ij})\ln 2}-\varphi_{ij}-p_{ij}(t)\right\}+Q_i+\tilde{B}_i(t)P_i/r_i, \tag{40}$$

and the second order partial derivative of $U_{b_i}(t)$ with respect to $b_{ij}(t)$ is

$$\frac{\partial^2 U_{b_i}(t)}{\partial b_{ij}(t)^2}=-V_i\left\{\rho_i\frac{1}{(1+b_{i0})^2\ln 2}\right\}. \tag{41}$$

Obviously, $\frac{\partial^2 U_{b_i}(t)}{\partial b_{ij}(t)^2} < 0$ and $U_{b_i}(t)$ is convex with respect to $b_{ij}(t)$. According to the KKT conditions, we can get the global optimum $b_{ij}^M = \frac{\rho_i}{C} - 1$. If $b_{ij}^L < b_{ij}^M \leq b_{ij}^U$, then $b_{ij}^* = b_{ij}^M$; if $b_{ij}^M \leq b_{ij}^L$, then $b_{ij}^* = b_{ij}^L$; and if $b_{ij}^M > b_{ij}^U$, then $b_{ij}^* = b_{ij}^U$. □

B. Optimal Strategy for Sellers/edge-Cloud Servers

For each seller/edge-cloud server, we need to determine the optimal price (i.e., $p_{ij}(t)$) and computing resource allocation (i.e., $f_{ij}(t)$) according to buyers' requirements. Following (32) and (39), we obtain the first order partial derivative of $U_{s_j}(t)$ with respect to $p_{ij}(t)$ as

$$\frac{\partial U_{s_j}(t)}{\partial p_{ij}(t)} = b_{ij}^* - p_{ij}\frac{\rho_i}{C^2}\ln 2 + 2\psi_j\kappa_j b_{ij}^*\frac{\rho_i}{C^2}\frac{{\gamma_i}^2}{\tau}\ln 2, \quad (42)$$

and the second order partial derivative of $U_{s_j}(t)$ with respect to $p_{ij}(t)$ as

$$\frac{\partial^2 U_{s_j}(t)}{\partial {p_{ij}(t)}^2} = -\frac{2\rho_i(\ln 2)^2}{C^2} \times\{\frac{B}{C\ln 2} + \frac{\psi_j\kappa_j b_{ij}^*\frac{{\gamma_i}^2}{\tau}}{C} + \psi_j\kappa_j\frac{{\gamma_i}^2}{\tau}\frac{\rho_i}{C^2}\}, \quad (43)$$

where $B = \varphi_{ij}\ln 2 + \psi_j\kappa_j b_{ij}^*\frac{{\gamma_i}^2}{\tau}\ln 2 - \frac{\ln 2}{V_i}\left(Q_i + \tilde{B}_i(t)\frac{P_i}{r_i}\right)$.

For each seller, it is natural that the revenue $U_{s_j}(t)$ should be non-negative [42]. Letting $U_{s_j}(t) = 0$, we obtain the cost price of seller j as $p_{ij}^c(t) = \psi_j\kappa_j b_{ij}^*\frac{{\gamma_i}^2}{\tau}$, which means that the minimum acceptable selling price of the seller is $p_{ij}^c(t)$.

Theorem 3: If the selling price of seller j is greater than $p_{ij}^c(t)$, we have $\frac{\partial^2 U_{s_j}(t)}{\partial(p_{ij}(t))^2} < 0$.

Proof: Since the minimum acceptable price of server j is $p_{ij}^c(t)$, we have $p_{ij}^c(t) \leq p_{ij}(t)$. Then, we have $B = \varphi_{ij}\ln 2 + p_{ij}^c(t)\ln 2 - \frac{\ln 2}{V_i}\left(Q_i(t) + \tilde{B}_i(t)\frac{P_i}{r_i}\right) \leq C$. Since $C > 0$, we have $\varphi_{ij}\ln 2 + p_{ij}^c(t)\ln 2 > \frac{\ln 2}{V_i}\left(Q_i(t) + \tilde{B}_i(t)\frac{P_i}{r_i}\right)$. Thus we have $0 < \frac{B}{C} \leq 1$ and $\frac{\partial^2 U_{s_j}(t)}{\partial(p_{ij}(t))^2} < 0$. □

Since (25) is an affine function, $U_{s_j}(t)$ is convex with respect to $p_{ij}(t)$. The optimal strategies in (32) can be obtained by solving the Lagrangian Multiplier and KKT conditions as follows.

$$p_{ij}^*(t) = \frac{C^2}{\rho_i\ln 2}b_{ij}^*(t) + 2\psi_j\kappa_j b_{ij}^*(t)\frac{{\gamma_i}^2}{\tau}. \quad (44)$$

$$f_{ij}^*(t) = \frac{b_{ij}^*(t)\gamma_i}{\tau_i^d}. \quad (45)$$

C. Existence of the Stackelberg Equilibrium

In this section, we prove that the optimal solution $\left(b_{ij}^*(t), p_{ij}^*(t)\right), i \in \mathcal{M}, j \in \mathcal{N}, t \in \mathcal{T}$ is the Stackelberg Equilibrium (SE) solution. For simplicity, we analyze the optimal offloaded task ($b_{ij}(t)$) and price ($p_{ij}(t)$) solutions in one time slot, which can be extended to other time slots easily. First, the SE of the proposed game is defined in the following.

Definition 1: $\left(b_{ij}^{\mathrm{SE}}(t), p_{ij}^{\mathrm{SE}}(t)\right)$ is an SE solution when the price $p_{ij}(t)$ of seller is determined, and $b_{ij}^{\mathrm{SE}}(t)$ satisfy

$$U_{b_i}\left(b_{ij}^{\mathrm{SE}}(t)\right) = \sup_{b_{ij}^{\max} \leq b_{ij}(t) \leq b_{ij}^{\min}} \{U_{b_i}(b_{ij}(t))\}, \quad \forall t \in \mathcal{T}, \quad (46)$$

and when the offloaded task $b_{ij}(t)$ is determined, and $p_{ij}^{\mathrm{SE}}(t)$ satisfy

$$U_{s_j}\left(p_{ij}^{\mathrm{SE}}(t)\right) = \sup_{p_{ij}(t) \geq p_{ij}^c} \left\{U_{s_j}(p_{ij}(t))\right\}, \quad \forall t \in \mathcal{T}. \quad (47)$$

Next we prove that the optimal solution $\left(b_{ij}^*(t), p_{ij}^*(t)\right)$ is $\left(b_{ij}^{\mathrm{SE}}(t), p_{ij}^{\mathrm{SE}}(t)\right)$.

Lemma 1: If the price $p_{ij}(t)$ of the seller/edge-cloud server is fixed, the revenue function $U_{b_i}(b_{ij}(t))$ of buyer/MD takes the maximum value at $b_{ij}^*(t)$.

Proof: ***Proposition 2*** shows that problem $\mathcal{P}2 - buyer''$ is convex with respect to $b_{ij}(t)$. Thus the revenue function $U_{b_i}(s_{ij}(t))$ assumes its maximum value at $b_{ij}^*(t)$. According to ***Definition 1***, $b_{ij}^*(t)$ is the SE $b_{ij}^{\mathrm{SE}}(t)$. □

Lemma 2: For buyers, the optimal offloaded task $b_{ij}^*(t)$ decreases with the increased seller's price $p_{ij}(t)$.

Proof: According to (39), we have $\frac{\partial b_{ij}^*(t)}{\partial p_{ij}(t)} = -\frac{\rho_i}{C^2}\ln 2 < 0$. Hence, $b_{ij}^*(t)$ is a monotonously decreasing function of $p_{ij}(t)$. It means that if the seller's price increases, the buyers will be more reluctant to buy, resulting in little or no revenue for seller. Thus, the sellers should adopt a suitable price. We obtain the seller's optimal price by solving $\frac{\partial U_{s_j}(p_{ij}(t))}{\partial p_{ij}(t)} = 0$. □

Lemma 3: If the optimal offloaded task $b_{ij}^*(t)$ of buyer/MD i is fixed, the $U_{s_j}(p_{ij}(t))$ of seller takes the maximum value at $p_{ij}^*(t)$.

Proof: According to ***Theorem 2***, problem $\mathcal{P}2 - seller'$ is convex with respect to $p_{ij}(t)$. Thus the revenue function $U_{s_j}(p_{ij}(t))$ takes the maximum value at $p_{ij}^*(t)$. According to ***Definition 1***, $p_{ij}^*(t)$ is the SE $p_{ij}^{\mathrm{SE}}(t)$. □

In summary, $\left(b_{ij}^*(t), p_{ij}^*(t)\right)$ is the optimal task offloading and price decisions, and it is also the SE solution $\left(b_{ij}^{\mathrm{SE}}(t), p_{ij}^{\mathrm{SE}}(t)\right)$.

Fig. 2 illustrates the relationship and interaction among the optimizations of task offloading, on-demand computing resource allocation, and battery energy management at each time slot. First, each MD/buyer needs to determine the energy harvesting decision on the basis of the sign of the virtual energy queue $\tilde{B}_i(t)$. After that, the task offloading and computing resource allocation strategies will be obtained among MDs/buyers and servers/sellers. Each MD/buyer decides task offloading $b_{ij}(t)$ and local execution $f_{i0}(t)$ strategies according to the announced market price $p_{ij}(t)$ of servers/sellers. Next, each server/seller updates selling price and allocates computing resource for heterogeneous MDs/buyers, and then MDs/buyers update their strategies. Note that several iterations may be necessary in the game to achieve convergence to the SE solutions $\left(b_{ij}^{\mathrm{SE}}(t), p_{ij}^{\mathrm{SE}}(t)\right)$. Finally, the servers/sellers can allocate on-demand computing resource to heterogeneous MDs through different equilibrium price $p_{ij}^{\mathrm{SE}}(t)$.

VI. Offloading Pre-Screening Criterion and Price Update Strategy Design

In this section, we develop an offloading pre-screening criterion to reduce the unnecessary communication signaling overhead and improve the efficiency of task processing. And then, the price update strategies of the sellers are designed.

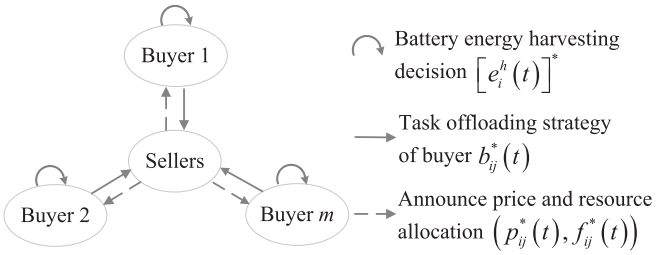


Fig. 2. Relationship and interaction among the optimizations in the market.

A. Offloading Pre-Screening Criterion

Heterogeneous mobile devices (MDs) have different battery energy level, offloading requirements, and traffic characteristics (e.g., task types and computation density). Moreover, since edge-cloud servers have different computing resources (e.g., availability of computing resource and computing costs) and are at different locations, the servers will set different prices to serve different MDs, which may not be appropriate for each MD. To reduce the unnecessary communication signaling overhead, it is essential that each MD selects one or more appropriate edge-cloud servers at the beginning of each time slot. There are two main factors that affect the MDs' offloading selection decisions: (i) the **b**attery discharging constraint factor (**B**) and (ii) the **p**rice factor (**P**).

Criterion B: This is the battery discharge constraint of each MD (11). We derive the minimum offloaded task in each time slot as follows.

$$b_{ij}^L(t) = \min\left\{f_j^{\min}\tau/\gamma_i,\ (E_i^{\min} - D)r_i/P_i(t)\right\}, \quad (48)$$

where $D = \kappa_i[f_{i0}^*(t)]^2\tau - \sum_{\forall x\in\mathcal{N}, x\neq j} P_i(t)b_{ix}^*(t)/r_i$. Accordingly, the maximum offloaded task in each time slot as follows.

$$b_{ij}^U(t) = \min\left\{Q_i^L(t), (\min\left\{E_i^{\max}, B_i(t)\right\} - D)r_i/P_i(t)\right\}, \quad (49)$$

where $Q_i^L(t) = Q_i(t) - b_{i0}(t) - \sum_{\forall x\in\mathcal{N}, x\neq j} b_{ix}^*(t)$.

If and only if $b_{ij}^L(t) \leq b_{ij}^U(t)$, it is feasible for MD i to offload task to edge-cloud server j. Otherwise, the edge-cloud server j will be excluded for MD i.

Criterion P: Due to different prices at the edge-cloud servers, each MD should first choose (or exclude) those more (or less) beneficial servers[6] and determine how many tasks to offload.

According to (40) and letting $\frac{\partial U_{b_i}(t)}{\partial b_{ij}(t)}|_{b_{ij}(t)=b_{ij}^{\min}} > 0$, we have

$$p_{ij}(t) < \frac{\rho_i}{\left(1+b_{ij}^{\min}\right)\ln 2} - \varphi_{ij} + \frac{Q_i(t) + \tilde{B}_i(t)\frac{P_i}{r_i}}{V_i}. \quad (50)$$

That is, if the quote price of edge-cloud server j satisfies inequality (50), the MD will obtain a large revenue U_{b_i} by increasing b_{ij}. When $b_{ij}(t) = b_{ij}^{\min}$, it is beneficial for MD i to offload tasks to edge-cloud server j. Otherwise, the edge-cloud server j will be excluded for MD i.

[6]it is natural that each MD chooses edge-cloud servers by observing how U_{b_i} varies with b_{ij}, i.e., the sign of $\frac{\partial U_{b_i}(t)}{\partial b_{ij}(t)}$.

Furthermore, ***Theorem 3*** indicates that the floor price of seller j for buyer i is $p_{ij}^{\min} = \psi_j\kappa_j b_{ij}^{\min}\gamma_i^2/\tau$. It follows that $\psi_j\kappa_j b_{ij}^{\min}\frac{\gamma_i^2}{\tau} < \frac{\rho_i}{(1+b_{ij}^{\min})\ln 2} - \varphi_{ij} + \frac{Q_i(t)+\tilde{B}_i(t)\frac{P_i}{r_i}}{V_i}$. Thus we have

$$\frac{Q_i(t) + \tilde{B}_i(t)\frac{P_i}{r_i}}{V_i} > \frac{\rho_i}{\left(1+b_{ij}^{\min}\right)\ln 2} - \varphi_{ij} - \psi_j\kappa_j b_{ij}^{\min}\frac{\gamma_i^2}{\tau}. \quad (51)$$

At the beginning of each time slot, due to constant $Q_i(t)$ and $\tilde{B}_i(t)$, the number of available servers is related to the control parameter V_i. The smaller the V_i, the more servers available.

B. Price Update Strategy

We first define the rth quoted price of seller j to buyer i as $p_{ij}^r(t)$ in time slot t. When the quote price strategies of all sellers are determined, each buyer decides task processing strategies (executed locally or offloaded) based on (36) and (39). According to ***Lemma 1***, we know that the SE solutions of buyers are $b_{ij}^*(t)$, for all $i \in \mathcal{M}$ and for all $j \in \mathcal{N}$.

Once the buyers reach the SE solutions, the sellers adjust price strategies based on the computation requirements of buyers. And the price update rate of sellers can be expressed as the Marginal Utility (MU) [44]. The price iteration process can be expressed as follows.

$$p_{ij}^{r+1}(t) = p_{ij}^r(t) + \nu\frac{\partial U_{s_j}(t)}{\partial p_{ij}^r(t)}, \quad (52)$$

where ν is the step size of price iteration, and $\frac{\partial U_{s_j}(t)}{\partial p_{ij}^r(t)} = b_{ij}^*(t) + p_{ij}^r(t)\frac{\partial b_{ij}(t)}{\partial p_{ij}^r(t)} - 2\psi_j\kappa_j\frac{b_{ij}^*(t)\gamma_i^2}{\tau}\frac{\partial b_{ij}(t)}{\partial p_{ij}^r(t)}$. To obtain the $(r+1)$th quote price of equation (50), each seller needs to receive feedback information $b_{ij}^*(t)$ and $\frac{\partial b_{ij}^*(t)}{\partial p_{ij}^r(t)}$ from the buyers.

For each seller, the revenue increases with the increase of quote price. Nevertheless, according to ***Definition 1*** and ***Lemma 3***, $U_{s_j}(t)$ is a convex function of $p_{ij}(t)$, which means that the seller cannot further increase the price when $\frac{\partial U_{s_j}(p_{ij}(t))}{\partial p_{ij}(t)} = 0$. Thus the seller price will reach the SE under the aforementioned constraints. Based on the above analysis, when all the sellers reach the SE, all of the buyers will reach the SE as well.

C. Computational Complexity Analysis

In the proposed scheme, the computational complexity is affected by the number of available servers/sellers for buyers, and the number of iterations among the buyers and sellers. Let $\tilde{n}_i$ denote the number of available servers for MD i, and Γ_{is} denote the number of iterations between MD i and server $s, s \in [0, \tilde{n}_i]$. For each MD and one suitable server (e.g., MD i and server s), the computational complexity is $O(\Gamma_{is})$. Overall, the computational complexity of the proposed scheme is $O\left(\sum_s \Gamma_{is}\right)$.

VII. Performance Evaluation

To evaluate the effectiveness and rationality of the proposed online **D**istributed **O**ffloading and **C**omputing resource

management with **E**nergy harvesting (DOCE) scheme for MEC-enabled heterogeneous IoT, we design three sets of simulations. First, we simulate the iterations of price update and the convergence speed in each time slot, and show the dynamic transaction prices in different time slots. Next, we show the differentiated and on-demand resource allocation for heterogeneous mobile devices (MDs). Finally, we conduct a comparison study with three benchmark schemes.

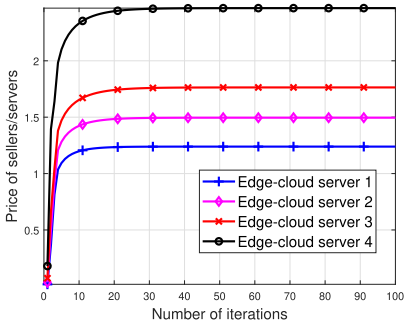
A. Simulation Setting

In the following simulations, we consider the scenarios with multiple MDs/buyers and multiple edge-cloud servers/sellers. For each MD, the task computation density γ_i is uniformly distributed between 800 cycle/Mbit and 1500 cycle/Mbit, and the arrival rate λ_i is distributed between 2 Mbit/s and 75 Mbit/s. The average energy harvesting power P_i^h (e.g., Piezoelectric generators [29]) is distributed between 2 mW to 50 mW, and harvested energy is uniformly distributed between 0 and $\delta_i^{\max} = P_i^h \tau$. The minimum and maximum battery dischargesare $E_i^{\min} = 0.01$ mJ and $E_i^{\max} = 1$ J in each time slot, which correspond to the Li-Ion batteries [45]. For each edge-cloud server, the minimum and maximum CPU frequency are $f_j^{\min} = 0.015$ GHz and $f_j^{\min} = 4$ GHz, respectively. In addition, similar to [6] and [7], we set $\rho_i = 2$, $f_i^{\min} = 0.01$ GHz, $f_i^{\max} = 2$ GHz, $\kappa_i = 1 \times 10^{-6} \sim 2 \times 10^{-7}$, $V_i = 100 \sim 1000$, $\varphi_i = 0.06 \sim 0.24$, $\kappa_j = 1 \times 10^{-7} \sim 10 \times 10^{-7}$, $\psi_j = 1$, $\tau = 1s$, and $P_i = 2$ mW. Moreover, the price update step is set to 5×10^{-5}, and each measurement is averaged over 20 instances. All the simulations are performed on a Workstation PC with Intel E5-2630 2.4GHz and 32GB memory.

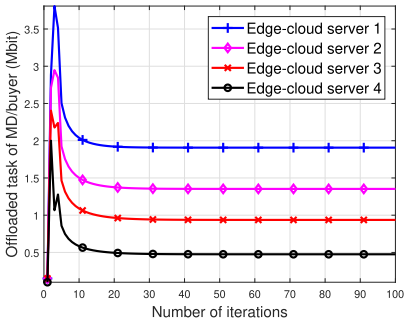
B. Dynamic Price Update and Computing Resource Allocation of the Proposed Scheme

In this simulation, we illustrate the iteration of the quote price and offloaded task updates in the proposed game. For easy observation, we consider one MD (i.e., $m = 1$) and four suitable edge-loud servers (i.e., $n = 4$) at different distances with $l_{ij,j=1,2,3,4} = 2, 4, 6, 10$ m, $\kappa_j = 1 \times 10^{-7}, 2 \times 10^{-7}, 4 \times 10^{-7}, 8 \times 10^{-7}$, $\lambda_i = 10$ Mbit/s, and $V_i = 100$.

1) Price Update and Convergence Speed: Fig. 3 shows the price updates at the edge-cloud servers/sellers and the offloaded task at the MD/buyer, as well as the convergence speed of the proposed scheme. In Fig. 3(a), the prices have a big initial increase and then gradually increase with iterations, finally approaching the optimal strategies ($p_{ij}^*(t)$). In each iteration, sellers check whether $\frac{\partial U_{s_j}(t)}{\partial p_{ij}(t)} > 0$ or not. If yes, the pricing is updated; otherwise, the procedure is terminated. The number of iterations until convergence is about 25. It can be seen from Fig. 3(b) that, since the initial price (p_{ij}^c) of sellers are lower, the offloaded task of the buyer first increases and then quickly decreases along iterations, eventually converging to the optimal value. Moreover, Fig. 3 illustrates that the bigger the communication distance and κ_j, the higher the optimal price. We also observe the opposite trend for offloaded tasks. This is because that the unit cost of processing tasks increases with the communication distance



(a) Price update



(b) Offloaded task update

Fig. 3. The evolution of the seller prices and the offloaded tasks.

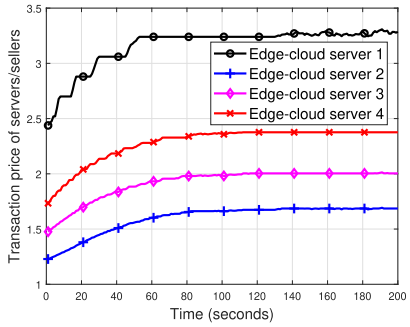
and κ_j, and the sellers have to maximize their revenue at a higher price. Accordingly, the buyer is likely to purchase more computing resource and offload more tasks to the edge-cloud servers with a lower price to maximize its revenue. According to Section V-C the optimal prices and the offloaded task are also SE solutions.

2) Dynamic Transaction Prices and Offloaded Tasks: Fig. 4 shows the evolution of the transaction prices of the servers/sellers and the offloaded tasks of the MD/buyer. Fig. 4(a) shows that the transaction prices among the buyer and sellers are gradually increasing until it stabilizes. And the offloaded tasks exhibit the same trend in Fig. 4(b). This is because that the queue backlog and battery energy level increase progressively before they become stable (as can be seen from the simulation results in the next subsection). The MD has more sufficient energy to process or offload task as time increases. Simultaneously, to ensure the stability of the task queue, more tasks need to be processed, i.e., the buyer needs to buy more computing resource from the sellers. When the battery energy level and task queue become stable, the transaction price and offloaded task do not increase further.

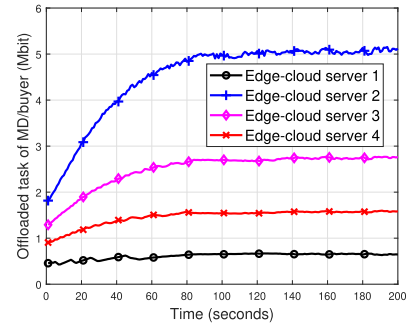
C. Differentiated Resource Allocation of the Proposed Scheme

In this simulation, we consider three sets of heterogeneous MDs ($m = 3$) with different computation density and latency requirements, and set $V_1 = 100$, $\gamma_1 = 1500$, $\lambda_1 = 10$Mbit/s, $V_2 = 300$, $\gamma_2 = 1000$, $\lambda_2 = 8$Mbit/s, and $V_3 = 600$, $\gamma_3 = 800$, $\lambda_3 = 5$Mbit/s,. The edge-cloud servers are the same as that in Section VII-B.

Fig. 5 shows the battery energy level, the task queue backlog, the revenue of the buyers, and the average computing



(a) Transaction price of servers/sellers



(b) Offloaded task of MD/buyer

Fig. 4. Transaction price and offloaded task versus time.

resource allocation over time. Fig. 5(a) and Fig. 5(b) show that both the battery energy level and task queue backlog achieve stability in the long-time evolution. The results also validate the feasibility of the system. Moreover, the battery energy level of each MD is stable at the perturbed energy level $\theta_i + \delta_i^{\max}$.

As we can see from the simulation results, since different devices have different requirements and data characteristics, they have different chronological order to reach stability. For example, because MD_1 has a larger task arrival rate and a smaller task cache demand, the task queue backlog of MD_1 will converge faster than other devices. Correspondingly, the battery energy level, revenue, and average resource allocated to MD_1 also become stable first. As shown in Fig. 5(c) and Fig. 5(d), due to the different requirements and characteristics of the MDs, different MDs are allocated with different computing resources. For example, since MD_1 has a higher latency sensitivity and greater computational density than others, MD_1 requests, and is allocated with more computing resource at the cost of revenue.

D. Comparison Study

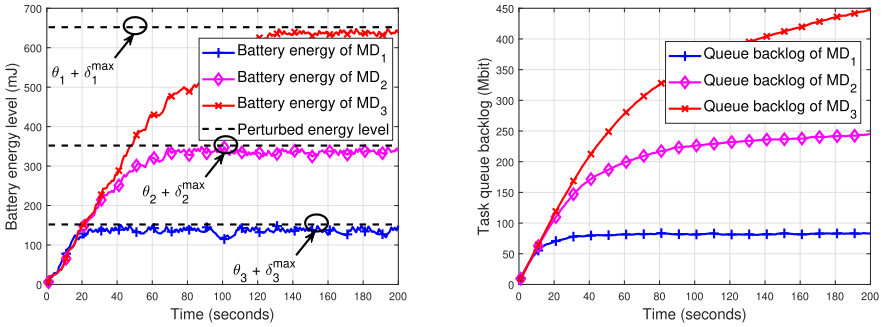
In this simulation, we conduct experiments to evaluate the performance in terms of system revenue, average energy efficiency (i.e., the processed task divided by system energy consumption), battery energy, and task queue backlog with the following three benchmark schemes.

- The ***Local-only*** scheme processes all tasks locally. Note that this scheme disables computation offloading, i.e., $b_{ij}^*(t) = 0$. The minimum and maximum CPU frequency are $f_{i0}^L(t) = \max\left\{f_{i0}^{\min}, \sqrt{\frac{E_i^{\min}}{\kappa_i \tau}}\right\}$ and $f_{i0}^U(t) = \min\left\{f_{i0}^{\max}, \sqrt{\frac{\min\left(E_i^{\max}, B_i(t)\right)}{\kappa_i \tau}}\right\}$, respectively. This allows us to demonstrate the improvement effects by invoking computation offloading.
- The ***Dynamic Naive Offloading (DNO)*** scheme is to let the MDs do their best to process task locally or offload to the optimal edge-cloud server in a centralized manner [6]. We set $f_{i0}^*(t) = \left(\frac{V_i}{-2\tilde{B}_i(t)\kappa_i}\right)^{\frac{1}{3}}$ and $b_{ij}^*(t) = \min\left\{Q_i(t) - b_{i0}^*(t), \frac{\min\{E_i^{\max}, B_i(t)\} - \kappa_i [f_{i0}^*(t)]\tau}{P_i(t)} r_i\right\}$.
- The ***Dynamic Greedy Offloading (DGO)*** scheme proposed in [7] processes computation tasks to maximize the system revenue in a centralized manner. Compared with this scheme, we can demonstrate the performance of the differentiated and on-demand dynamic computing resource allocation for heterogeneous MDs.

1) Performance Versus Time: Fig. 6 shows the performance comparison of average revenue, energy efficiency, battery energy level, and task queue backlog over long-term evolution. We can see from Fig. 6(a) and Fig. 6(c) that the average system revenue and battery energy level gradually increase initially. This is because that the processed task gradually increases as the backlog of the task queue increases. After that, since the processed task becomes stable or reaches the maximum, the average system revenue and the battery energy level reach stable values. Fig. 6(b) and Fig. 6(d) show that although the DNO scheme has a lower task queue backlog at the beginning, its performance is inadequate in the long-term due to its simple resource allocation method. Moreover, we can see that the DOCE scheme, with a differentiated and flexible computing resource allocation algorithm, outperforms the other three schemes significantly in average system revenue, energy efficiency, and task queue latency.

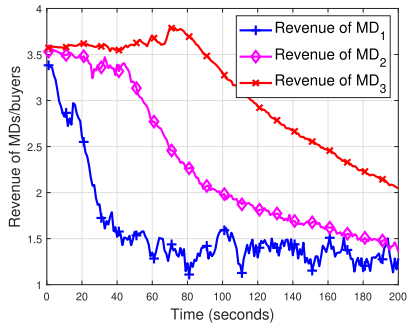
2) Varying Task Arrival Rate: Fig. 7 presents a performance comparison of the four schemes under increased task arrival rate, including the system revenue, the energy efficiency, the battery energy level, and the task queue backlog. We can see from Fig. 7(a), Fig. 7(b), and Fig. 7(c) that the average system revenue, the energy efficiency, and the maximum battery energy level all gradually decrease with the increased task arrival rate. This is because that as the task arrival rate is increased, there are more backlog in the task queue, as can be seen from Fig. 7(d), and there are more tasks need to be processed locally or offloaded at the expense of revenue, energy efficiency, and battery energy. Moreover, the overall performance of the proposed scheme is significantly better than the DNO and DGO schemes.

3) Varying Energy Harvesting Capability: Fig. 8 presents our performance comparison under varying energy harvesting capability of the MDs. As we can see from Fig. 8(a) and Fig. 8(c), the average system revenue and the maximum battery energy level gradually increase and then converge to stable values with the increase of the average energy harvesting power. The average energy efficiency and the maximum task queue backlog gradually decrease, as shown in Fig. 8(b) and Fig. 8(d). This is because that with the increase of energy harvesting power, the MDs have more battery energy and will

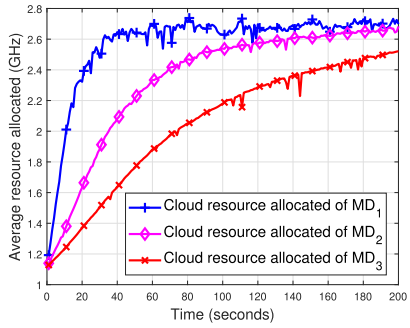


(a) Battery energy level

(b) Task queue backlog

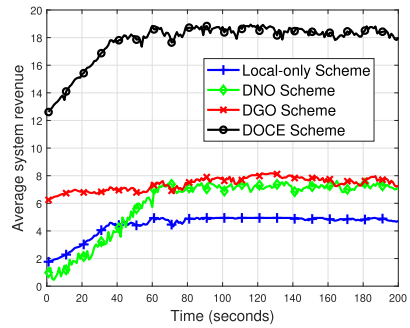


(c) Revenue of MDs/buyers

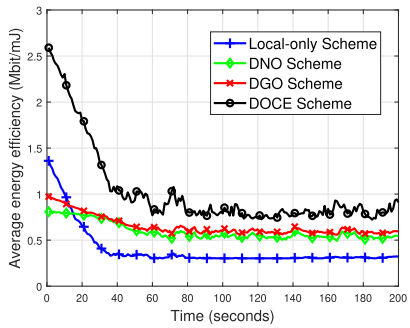


(d) Average resource allocated

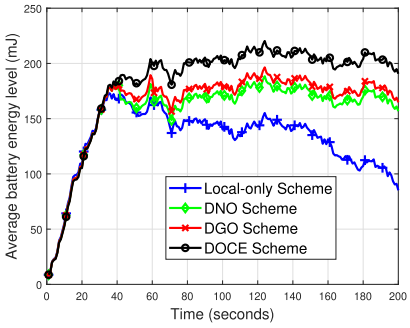
Fig. 5. Battery energy level, queue backlog, revenue, and resource allocation versus time.



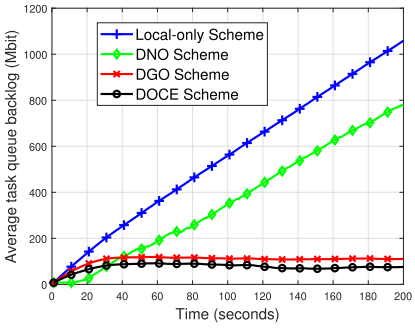
(a) Average system revenue



(b) Average energy efficiency



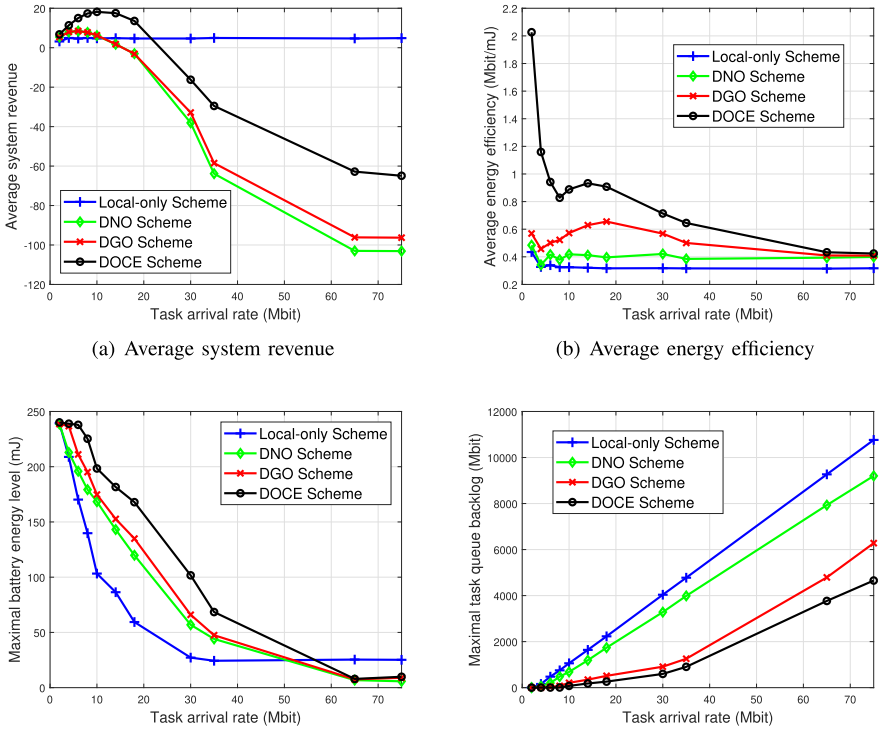
(c) Average battery energy level



(d) Average task queue backlog

Fig. 6. Performance comparison of average revenue, energy efficiency, battery energy level, and task queue backlog over time ($\lambda = 10Mbit/s$ and $P^h = 10mW$).

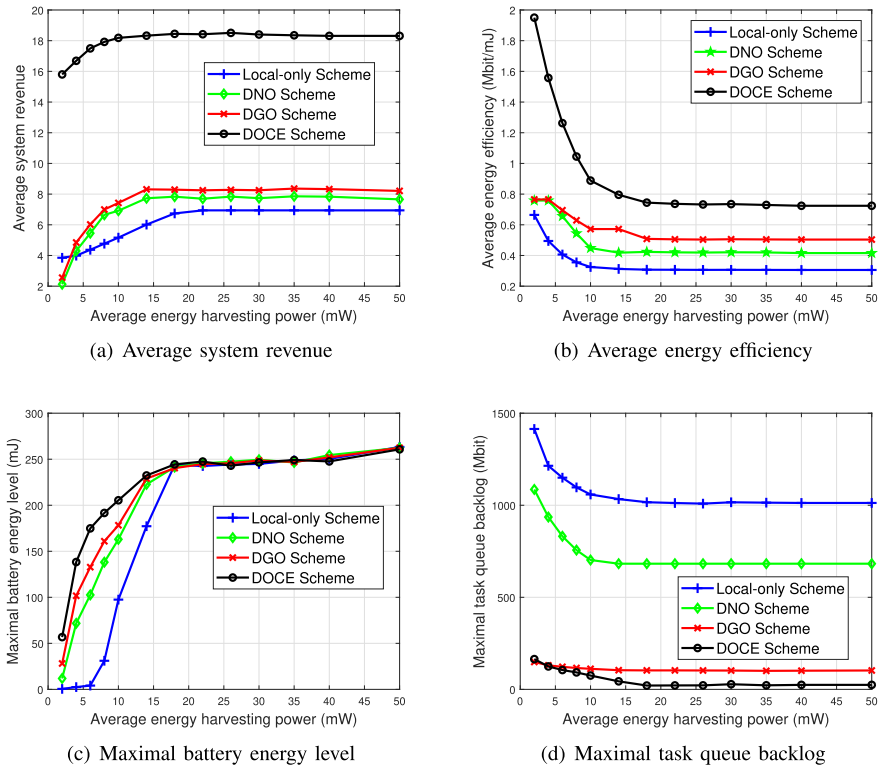
prefer to process more tasks locally or offload them to the edge servers. As the energy harvesting power exceeds 20 mW, the renewable harvested energy in real time is sufficient to support the need of each MD. The revenue and energy efficiency of the proposed scheme are both significantly higher than that of the DNO and DGO schemes.



(a) Average system revenue (b) Average energy efficiency

(c) Maximal battery energy level (d) Maximal task queue backlog

Fig. 7. System performance under increased task arrival rate ($P^h = 10mW$).

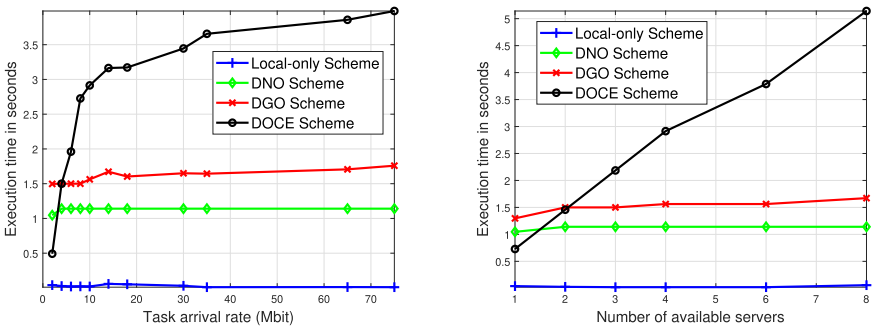


(a) Average system revenue (b) Average energy efficiency

(c) Maximal battery energy level (d) Maximal task queue backlog

Fig. 8. Performance comparison under varying energy harvesting power ($\lambda = 10Mbit/s$).

4) Execution Time: Fig. 9 shows the execution time of the schemes under different tasks arrival rates and numbers of available servers. We can observe that the execution time of the DOCE scheme increases with the increase of task arrival rate in Fig. 9(a). The reason is that the larger the offloaded task, the more iterations between each buyer and



(a) Varying task arrival rate ($m = 2, n = 4$) (b) Varying available servers ($m = 2, \lambda = 10Mbit/s$)

Fig. 9. Execution time of the different schemes.

seller to reach the SE solution $\left(b_{ij}^{\mathrm{SE}}(t), p_{ij}^{\mathrm{SE}}(t)\right)$. We can also see from Fig. 9(b) that the execution time increases when more servers are available. This is because each buyer will play games with more available sellers/servers to obtain the optimal offloading strategy. Although the execution times of the Local-ofnly scheme, DNO scheme, and DGO scheme remain almost unchanged in the simulations, this is based on the assumption that the network state information is perfectly and fully known in advance for each MD and server, which is an ideal assumption in a dynamic and random wireless network.

VIII. Conclusion

In this paper, we investigated a distributed EH-enabled MEC offloading system for heterogeneous IoT, and jointly optimized task offloading, computing resource allocation, and battery energy management based on game-theoretic and perturbed Lyapunov optimization theory. First, this paper developed heterogeneous task offloading strategies as well as achieving the stability of the battery energy level and guaranteed the computation performance in the long-term evolution. Moreover, aiming to allocate limited edge-cloud computing resource on demand, a dynamic quote price mechanism of edge-cloud resource was designed based on different computational requirements. To reduce the unnecessary communication signaling overhead and choice the suitable edge-cloud servers, an offloading pre-screening criterion was proposed by balancing battery energy level, computation latency, and revenue. The numerical results validated that the proposed distributed offloading scheme outperforms three baseline schemes. An unique advantage of the proposed scheme is that it works online, only requiring the current system state without the overhead of distributing computation task requests, cloud available resource, wireless channel state, and EH processes.

Since the computing resource of edge-cloud servers is finite, they may not be able to respond quickly to bursty computation requirements. The queueing delay could be high for computation-intensive tasks. In our future work, we will address the queueing delay in the cloud. Moreover, extensive experiments will be conduct to verify the effectiveness and reliability of the proposed algorithm in a realistic system in the future.

Appendix A
Proof of Theorem 1

Proof: According to the energy harvesting optimal strategy (34), only if $\tilde{B}_i(t) \leq 0$, namely, $B_i(t) \leq \theta_i$, the MD i needs to harvest energy δ_i^a ($0 \leq \delta_i^a(t) \leq \delta_i^{\max}$). Thus, we have $0 \leq B_i(t) \leq \theta_i + \delta_i^{\max}$ in time slot t. Next, we show that $B_i(t+1) \leq \theta_i + \delta_i^{\max}$ in the following.

1) If $\tilde{B}_i(t) > 0$, i.e., $B_i(t) > \theta_i$, MD i will not harvest energy and thus $[e_i^a(t)]^* = 0$. We have $B_i(t+1) \leq B_i(t) + \delta_i^{\max} \leq \theta_i + \delta_i^{\max}$.
2) If $\tilde{B}_i(t) \leq 0$, i.e., $B_i(t) \leq \theta_i$, MD i will harvest energy and thus $[e_i^a(t)]^* = \delta_i(t)$. We also have $B_i(t+1) \leq B_i(t) + \delta_i^{\max} \leq \theta_i + \delta_i^{\max}$.

Following the iteration relationship of $B_i(t)$, we thus have that the battery energy level of MD i always satisfies $0 \leq B_i(t) \leq \theta_i + \delta_i^{\max}$ in each time slot t. □

Appendix B
Proof of Theorem 2

Proof: Since $b_{ij}(t) \geq 0$, $a_i(t) \geq 0$, $e_i^h(t) \geq 0$ and $e_{i0}(t) \geq 0$, we have the following inequality.

$$\begin{aligned}
&L\left[\Theta_i(t+1)\right] \\
&= \frac{1}{2}\left\{\left(Q_i(t)\right)^2 + 2Q_i(t)\left(a_i(t) - b_i(t)\right)\right\} \\
&\quad + \frac{1}{2}\left\{\left(a_i(t)\right)^2 + \left(b_i(t)\right)^2\right\} \\
&\quad + \frac{1}{2}\left\{\left(B_i(t)\right)^2 + 2B_i(t)\left(e_i^h(t) - e_{i0}^t(t)\right)\right\} \\
&\quad + \frac{1}{2}\left\{\left(e_i^h(t)\right)^2 + \left(e_{i0}^t(t)\right)^2\right\} \\
&\leq \frac{1}{2}\left(Q_i(t)\right)^2 + Q_i(t)\left(a_i(t) - b_i(t)\right) \\
&\quad + \frac{1}{2}\left(B_i(t)\right)^2 + B_i(t)\left(e_i^h(t) - e_{i0}^t(t)\right) + \Phi_i, \quad (53)
\end{aligned}$$

where $\Phi_i = \frac{1}{2}(a_i^{\max})^2 + \frac{1}{2}(b_i^{\max})^2 + \frac{1}{2}\left(\delta_i^h\right)^2 + \frac{1}{2}(E_i^{\max})^2 \geq 0$. According to inequality (53), we can deduce the follow inequality.

$$\begin{aligned}
\Delta\left[\Theta_i(t)\right] \leq\ & \mathbb{E}\left\{Q_i(t)\left(a_i(t) - b_i(t)\right)\right\} \\
& + \mathbb{E}\left\{\tilde{B}_i(t)\left(e_i^h(t) - e_{i0}^t(t)\right)\right\} + \Phi_i. \quad (54)
\end{aligned}$$

Adding $-V_i\mathbb{E}\left(u_{b_i}\left(t\right)|\Theta_i\left(t\right)\right)$ on both sides of (54), we have the inequality (30). □

REFERENCES

[1] S. Xia, Z. Yao, and Y. Li, "A distributed stochastic task offloading methodology for IoT on e-Health," in *Proc. IEEE ICC*, Dublin, Ireland, Jun. 2020, pp. 1–6.

[2] *Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2018–2023*, Cisco, San Jose, CA, USA, Mar. 2020.

[3] P. Wang, Z. Zheng, B. Di, and L. Song, "HetMEC: Latency-optimal task assignment and resource allocation for heterogeneous multi-layer mobile edge computing," *IEEE Trans. Wireless Commun.*, vol. 18, no. 10, pp. 4942–4956, Oct. 2019.

[4] T. X. Vu, S. Chatzinotas, B. Ottersten, and A. V. Trinh, "Full-duplex enabled mobile edge caching: From distributed to cooperative caching," *IEEE Trans. Wireless Commun.*, vol. 19, no. 2, pp. 1141–1153, Feb. 2020.

[5] D. Ma, G. Lan, M. Hassan, W. Hu, and S. K. Das, "Sensing, computing, and communications for energy harvesting IoTs: A survey," *IEEE Commun. Surveys Tuts.*, vol. 22, no. 2, pp. 1222–1250, 2nd Quart., 2020.

[6] 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.

[7] G. Zhang, W. Zhang, Y. Cao, D. Li, and L. Wang, "Energy-delay tradeoff for dynamic offloading in mobile-edge computing system with energy harvesting devices," *IEEE Trans. Ind. Informat.*, vol. 14, no. 10, pp. 4642–4655, Oct. 2018.

[8] Y. Zhang, J. He, and S. Guo, "Energy-efficient dynamic task offloading for energy harvesting mobile cloud computing," in *Proc. IEEE Int. Conf. Netw., Archit. Storage*, Chongqing, China, Oct. 2018, pp. 1–4.

[9] H. T. Dinh, C. Lee, D. Niyato, and P. Wang, "A survey of mobile cloud computing: Architecture, applications, and approaches," *Wireless Commun. Mobile Comput.*, vol. 13, no. 18, pp. 1587–1611, Dec. 2013.

[10] Y. Luo, L. Pu, Y. Zhao, W. Wang, and Q. Yang, "A nonlinear recursive model based optimal transmission scheduling in RF energy harvesting wireless communications," *IEEE Trans. Wireless Commun.*, vol. 19, no. 5, pp. 3449–3462, May 2020.

[11] R. Deng, R. Lu, C. Lai, T. H. Luan, and H. Liang, "Optimal workload allocation in fog-cloud computing toward balanced delay and power consumption," *IEEE Internet Things J.*, vol. 3, no. 6, pp. 1171–1181, Dec. 2016.

[12] S. Dai, M. L. Wang, Z. Gao, L. Huang, X. Du, and M. Guizani, "An adaptive computation offloading mechanism for mobile health applications," *IEEE Trans. Veh. Technol.*, vol. 69, no. 1, pp. 998–1007, Jan. 2020.

[13] B. Liu, C. Liu, and M. Peng, "Resource allocation for energy-efficient MEC in NOMA-enabled massive IoT networks," *IEEE J. Sel. Areas Commun.*, vol. 39, no. 4, pp. 1015–1027, Apr. 2021.

[14] C. You, K. Huang, H. Chae, and B.-H. Kim, "Energy-efficient resource allocation for mobile-edge computation offloading," *IEEE Trans. Wireless Commun.*, vol. 16, no. 3, pp. 1397–1411, Mar. 2017.

[15] J. Yang and S. Ulukus, "Optimal packet scheduling in an energy harvesting communication system," *IEEE Trans. Commun.*, vol. 60, no. 1, pp. 220–230, Jan. 2012.

[16] S. S. Gupta and N. B. Mehta, "Revisiting effectiveness of energy conserving opportunistic transmission schemes in energy harvesting wireless sensor networks," *IEEE Trans. Commun.*, vol. 67, no. 4, pp. 2968–2980, Apr. 2019.

[17] K. Li *et al.*, "Fair scheduling for data collection in mobile sensor networks with energy harvesting," *IEEE Trans. Mobile Comput.*, vol. 18, no. 6, pp. 1274–1287, Jun. 2019.

[18] M. Calvo-Fullana, J. Matamoros, and C. Antón-Haro, "Sensor selection and power allocation strategies for energy harvesting wireless sensor networks," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 12, pp. 3685–3695, Dec. 2016.

[19] S. Guo, Y. Shi, Y. Yang, and B. Xiao, "Energy efficiency maximization in mobile wireless energy harvesting sensor networks," *IEEE Trans. Mobile Comput.*, vol. 17, no. 7, pp. 1524–1537, Jul. 2018.

[20] Y. Hu, C. Qiu, and Y. Chen, "Lyapunov-optimized two-way relay networks with stochastic energy harvesting," *IEEE Trans. Wireless Commun.*, vol. 17, no. 9, pp. 6280–6292, Sep. 2018.

[21] F. Wang and X. Zhang, "Dynamic interface-selection and resource allocation over heterogeneous mobile edge-computing wireless networks with energy harvesting," in *Proc. IEEE INFOCOM (INFOCOM WKSHPS)*, Honolulu, HI, USA, Apr. 2018, pp. 190–195.

[22] Y. Teng, K. Cheng, Y. Zhang, and X. Wang, "Mixed-timescale joint computational offloading and wireless resource allocation strategy in energy harvesting multi-MEC server systems," *IEEE Access*, vol. 7, pp. 74640–74652, 2019.

[23] G. Zhang, Y. Chen, Z. Shen, and L. Wang, "Distributed energy management for multiuser mobile-edge computing systems with energy harvesting devices and QoS constraints," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4035–4048, Jun. 2019.

[24] Y. Zhuang, X. Li, H. Ji, and H. Zhang, "Optimization of mobile MEC offloading with energy harvesting and dynamic voltage scaling," in *Proc. IEEE WCNC*, Marrakesh, Morocco, Apr. 2019, pp. 1–6.

[25] A. Mahmood, A. Ahmed, M. Naeem, and Y. Hong, "Partial offloading in energy harvested mobile edge computing: A direct search approach," *IEEE Access*, vol. 8, pp. 36757–36763, 2020.

[26] X. Chen, H. Zhang, C. Wu, S. Mao, Y. Ji, and M. Bennis, "Optimized computation offloading performance in virtual edge computing systems via deep reinforcement learning," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4005–4018, Jun. 2019.

[27] F. Hao, G. Pang, Z. Pei, K. Qin, Y. Zhang, and X. Wang, "Virtual machines scheduling in mobile edge computing: A formal concept analysis approach," *IEEE Trans. Sustain. Comput.*, vol. 5, no. 3, pp. 319–328, Jul. 2020.

[28] X. Li *et al.*, "ViPSN: A vibration-powered IoT platform," *IEEE Internet Things J.*, vol. 8, no. 3, pp. 1728–1739, Feb. 2021.

[29] A. Khaligh, P. Zeng, and C. Zheng, "Kinetic energy harvesting using piezoelectric and electromagnetic technologies—State of the art," *IEEE Trans. Ind. Electron.*, vol. 57, no. 3, pp. 850–860, Mar. 2010.

[30] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2322–2358, 4th Quart., 2017.

[31] A. P. Miettinen and J. K. Nurminen, "Energy efficiency of mobile clients in cloud computing," in *Proc. USENIX Conf. Hot Topics Cloud Comput. (HotCloud)*, Boston, MA, USA, Jun. 2010, pp. 1–7.

[32] M. Andrews, A. F. Anta, L. Zhang, and W. Zhao, "Routing for energy minimization in the speed scaling model," in *Proc. IEEE INFOCOM*, San Diego, CA, USA, Mar. 2010, pp. 1–9.

[33] K. Son and B. Krishnamachari, "SpeedBalance: Speed-scaling-aware optimal load balancing for green cellular networks," in *Proc. IEEE INFOCOM*, Orlando, FL, USA, Mar. 2012, pp. 2816–2820.

[34] M. Gorlatova, A. Wallwater, and G. Zussman, "Networking low-power energy harvesting devices: Measurements and algorithms," *IEEE Trans. Mobile Comput.*, vol. 12, no. 9, pp. 1853–1865, Sep. 2013.

[35] M.-L. Ku, W. Li, Y. Chen, and K. J. R. Liu, "Advances in energy harvesting communications: Past, present, and future challenges," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 2, pp. 1384–1412, 2nd Quart., 2016.

[36] S. Lakshminarayana, T. Q. S. Quek, and H. V. Poor, "Cooperation and storage tradeoffs in power grids with renewable energy resources," *IEEE J. Sel. Areas Commun.*, vol. 32, no. 7, pp. 1386–1397, Jul. 2014.

[37] D. Niyato, E. Hossain, and P. Wang, "Optimal channel access management with QoS support for cognitive vehicular networks," *IEEE Trans. Mobile Comput.*, vol. 10, no. 4, pp. 573–591, Apr. 2011.

[38] T. Truong-Huu, C.-K. Tham, and D. Niyato, "A stochastic workload distribution approach for an ad hoc mobile cloud," in *Proc. IEEE CloudCom*, Singapore, Dec. 2014, pp. 174–181.

[39] B. Cao, S. Xia, J. Han, and Y. Li, "A distributed game methodology for crowdsensing in uncertain wireless scenario," *IEEE Trans. Mobile Comput.*, vol. 19, no. 1, pp. 15–28, Jan. 2020.

[40] M. J. Neely, *Stochastic Network Optimization With Application to Communication and Queueing Systems*, (Synthesis Lectures on Communication Networks). Williston, VT, USA: Morgan & Claypool, 2010.

[41] M. J. Neely and L. Huang, "Dynamic product assembly and inventory control for maximum profit," in *Proc. IEEE CDC*, Atlanta, GA, USA, Dec. 2010, pp. 2805–2812.

[42] B. Wang, Z. Han, and K. J. R. Liu, "Distributed relay selection and power control for multiuser cooperative communication networks using Stackelberg game," *IEEE Trans. Mobile Comput.*, vol. 8, no. 7, pp. 975–990, Jul. 2009.

[43] M. S. Barzaraa, *Nonlinear Programming: Theory and Algorithms*, 2nd ed. Hoboken, NJ, USA: Wiley, 1993.

[44] M. S. S. Rao and S. A. Soman, "Marginal pricing of transmission services using min-max fairness policy," *IEEE Trans. Power Syst.*, vol. 30, no. 2, pp. 573–584, Mar. 2015.

[45] S.-W. Luan, J.-H. Teng, D.-J. Lee, Y.-Q. Huang, and C.-L. Sung, "Charging/discharging monitoring and simulation platform for Li-ion batteries," in *Proc. TENCON, IEEE Region 10 Conf.*, Bali, Indonesia, Nov. 2011, pp. 868–872.