QoE Aware and Cell Capacity Enhanced Computation Offloading for Multi-Server Mobile Edge Computing Systems with Energy Harvesting Devices

Hailiang Zhao, Wei Du, Wei Liu, Tao Lei and Qiwang Lei

QoE Aware and Cell Capacity Enhanced Computation Offloading for Multi-Server Mobile Edge Computing Systems with Energy Harvesting Devices

Hailiang Zhao
hliangzhao97@gmail.com

Wei Du*, †
whutduwei@whut.edu.cn

Wei Liu*
wliu@whut.edu.cn

Tao Lei
leitao1995@whut.edu.cn

Qiwang Lei
leiqiwang@whut.edu.cn

College of Computer Science and Technology, Wuhan University of Technology, China
*Hubei Key Laboratory of Transportation Internet of Things, Wuhan, China

Abstract—The increasing complexity of intelligent services requires new paradigm to overcome the problems caused by resource-limited mobile devices. Mobile edge computing systems with energy harvesting devices is such a promising technology. By offloading the computation tasks from the mobile devices to the MEC servers, users could experience services with low latency. In addition, energy harvesting technology releases the tension between high energy consumption of intelligent services and capacity-constrained mobile device batteries. However, in multi-user and multi-server scenarios where mobile devices can move arbitrarily, computation offloading strategies are faced with new challenges because of resource competition and server selection. In this paper, we will develop an intelligent computation offloading strategy. The quality of user experience cost and the cell capacity in terms of the ratio of computation tasks offloaded will be adopted as the performance metrics. An online algorithm, namely, the LODCO-Based[1] Genetic Algorithm with Greedy Policy, will be proposed. Specifically, the algorithm is based on Lyapunov Optimization and the LODCO Algorithm. By choosing the execution mode among local execution, offloading execution and task dropping for each mobile device, our algorithm can asymptotically obtain the optimal results for the whole system. The algorithm proposed is low-complexity and could work without too much priori knowledge. Moreover, the algorithm not only inherits every advantage from the LODCO Algorithm but also adapts perfectly to the more complex environment. Simulation results illustrate that the algorithms could improve the ratio of offloading computation tasks by more than 10% while the QoE is guaranteed.

Keywords—*Intelligent services; Mobile edge computing; Energy harvesting; Computation offloading; Genetic algorithm; Greedy policy.*

I. INTRODUCTION

The rapid development of wireless communication technology and the increasing popularity of smart mobile devices are paving the way towards more intelligent services, such as virtual reality, augmented reality and face recognition. However, those services are usually computational-intensive, latency-sensitive and energy-consuming, which could not be supported by mobile devices with limited computational resources and energy provided by batteries [1]. Therefore, new paradigms are needed to carter to the requirements of intelligent services. Mobile edge computing (MEC) systems with energy harvesting (EH) devices is such a promising technology that IT services and cloud computing capability are provided within the radio access network [2]. By offloading the computation tasks from the mobile devices to the MEC servers, users could experience services with low latency [3]. Furthermore, energy consumption becomes not such an urgent problem because the mobile devices could be powered by free recycling energy sources [4].

The design of efficient computation offloading strategies has attracted research interest in recent years. Most works focus on single-user and single-server MEC systems with energy harvesting devices [5]. Nevertheless, for a system with multiple users and multiple servers, which are more typical scenarios in the real world [8], those strategies will not be applicable directly because of new challenges. To be specific, the computational resource and the radio resource are shared by multiple mobile devices. As a result, interference and competition could not be ignored. How to allocate limited resources among mobile devices should be investigated. Furthermore, a mobile device probably could connect to more than one MEC server. Then, the

mobile device could choose one server to offload its task. So, how to choose a proper server according to the system optimization metrics or the user's preference has to be studied. Additionally, as is known, the infrastructure providers will earn more profit with larger cell capacity for each base station [9], which is quantified by the ratio of computation tasks offloaded in this paper. On the other hand, more offloading means fiercer resource contention and thus worse quality of user experience (QoE). How to achieve the tradeoff between cell capacity enhancement and QoE guarantee will be a realistic problem.

In this paper, we design an intelligent computation offloading strategy for multi-user and multi-server MEC systems with EH devices. Our major contributions are summarized as follows:

- We consider a general MEC system with multiple EH mobile devices and multiple resource-constrained MEC servers where every mobile device can move arbitrarily within certain areas.

- User mobility and its effect on resources competition and server selection are embodied in the system model, which is formulated as a non-convex optimization problem finally.

- The quality of user experience cost (i.e. execution latency and penalty for dropped tasks) and the cell capacity in terms of the ratio of offloading computation tasks are optimized simultaneously.

- After applying Lyapunov Optimization, the original stochastic optimization problem is converted to a deterministic optimization problem at each time slot, which is a cornerstone of the proposed intelligent algorithm.

- The proposed algorithm, i.e., the LODCO-Based Greedy Algorithm with Greedy Policy, can deal with the correlation between any two mobile devices when choosing the computation modes, especially the offloading execution, which is out of the scope considered by the LODCO Algorithm.

- The proposed algorithm can obtain the optimal results after several iterations. By comparison with the LODCO Algorithm, our algorithm can not only keep perfectly the advantages of the LODCO Algorithm but also promote notably the ratio of offloading computation tasks while guaranteeing the QoE.

The organization of this paper is as follows. We survey state of the art in Section II. In section III, the system model will be introduced. In section IV, the LODCO-Based Genetic Algorithm with Greedy Policy will be proposed based on the formulated problem. Simulation results and Conclusion of this paper will be demonstrated in Section V and Section VI, respectively.

II. RELATED WORKS

Computation offloading for multi-user and multi-server mobile systems is a very challenging problem because of complexity of the scenario and interdependence among users and servers. A few strategies have proposed in recent years. In [10], the power-delay tradeoff in the context of task offloading was studied. The problem was formulated as a computation and transmits power minimization subject to latency and reliability constraints. In [11], the power minimization for the mobile devices by data offloading was investigated. Centralized and distributed algorithms for joint power allocation and channel assignment together with decision making were proposed. In [12], the problem of joint task offloading and resource allocation was formulated as a mix integer non-linear program. The task offloading decision, uplink transmission power of mobile users and computational resource allocation at the MEC servers were jointly optimized. The users' task offloading gains, which are measured by the reduction in task completion time and energy consumption, were maximized. Obviously, energy consumption is always optimized in all the works mentioned above. However, the optimization objective of MEC systems with EH devices is shifted from minimizing the battery energy consumption as the harvested energy is ample and free. As a result, those computation offloading strategies dedicated to the energy conservation cannot be utilized without modification.

In [13], a device-edge-cloud MEC system was investigated. A network aware multi-user and multi-edge computation partitioning problem was formulated. Computation and radio resources were allocated such that the average throughput of the users was maximized. The system considered in [13] is similar to this paper. Nevertheless, a partial offloading model was utilized in [13] while a binary offloading is exploited in our work.

There are works dedicated on the computation offloading for MEC systems with EH devices [5]. In [6], a "hotbooting" Q-learning based computation offloading scheme for IoT devices with energy harvesting to achieve the optimal offloading was presented. Also, a fast DQN-based IoT computation offloading scheme to compass the state space dimension was proposed. However, similar to [13], a partial offloading model was discussed in [6]. In [7], the problem of solving an optimal computation offloading policy was modeled as a Markov decision process, where the objective was to minimize the long-term cost. A deep Q-network-based strategic computation offloading algorithm was designed. Nevertheless, the system consisted of one telecom cloud, which was connected to multiple base stations. And the strategy was for base station selection. Obviously, both the system and the problem considered in [7] are different from that considered in our work. In [5], a green MEC system with EH devices was analyzed, and an effective computation offloading strategy was developed. Moreover, the execution cost, which includes both the execution latency and task failure, was adopted as the performance metric. Finally, a low-complexity online algorithm was proposed.

Although [5] is the most similar work to ours, several key differences should be addressed. Firstly, while the system discussed in [5] focused on a single user and single MEC server system, we dedicate on a multi-user and multi-server scenario. Secondly, user mobility, which was ignored in [5], is considered in this paper. Thirdly, the MEC server in this paper has limited computation capability, which generalizes the work in [5], where the MEC server was assumed to be with unlimited computational resources. Fourthly, compared to the single-objective optimization problem formulated in [5], we are interested in optimizing two objectives simultaneously. Due to

the differences mentioned above, our work is more complicated and difficult than that in [5].

III. SYSTEM MODEL

A. System Description

We consider a MEC system consisting of N mobile devices equipped with EH components and M MEC servers, where each mobile device and each MEC server share the same property, respectively. We assume that time is slotted, and denote the time slot length and the time slot index set by τ and $\mathcal{T} \triangleq \{0,1, \dots\}$, separately. We use $\mathcal{N} \triangleq \{1,2,\dots,N\}$ to denote the set of mobile devices and $\mathcal{M} \triangleq \{1,2,\dots,M\}$ to denote the set of MEC servers.
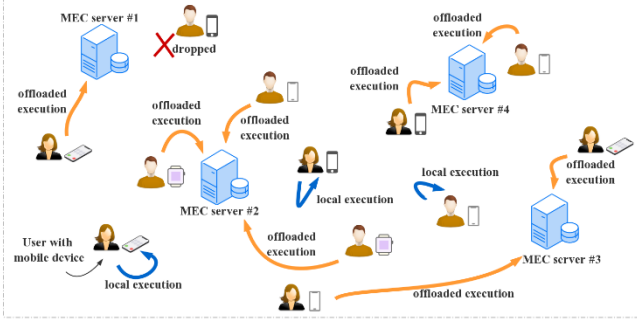

Fig. 1. The system with multiple mobile devices and multiple MEC servers.

As shown in Fig. 1, all mobile devices and MEC servers are limited to a specific area, where each MEC server is located at a particular position without moving and each mobile device can move around arbitrarily. Meanwhile, each MEC server is accessible to each mobile device. Besides, each mobile device's location is assumed to be independent and identically distributed (i.i.d.), i.e., the ith mobile device's location remains static within each time slot but varies among different time slots. We use $\mathcal{W}$ and $\mathcal{L}$ to denote the width and length of the specific area, respectively. Denote the distance between the ith mobile device and the jth MEC server at the tth time slot as $d_{i,j}^t$, and $d_{i,j}^t \sim U(0, d)$, $t \in \mathcal{T}$, $i \in \mathcal{N}$, $j \epsilon \mathcal{M}$, where $d \leq \sqrt{\mathcal{W}^2 + \mathcal{L}^2}$. i.e., the distance matrix $D^t \triangleq (d_{i,j}^t)_{N \times M}$ is deterministic at each time slot while could vary among different time slots.

B. Computation Task Model

We focus on delay-sensitive computation tasks with the execution deadline no greater than the length of each time slot [10]. Denote the computation task generated by the ith mobile device at the tth time slot as CT_i^t, which has the fixed size L (in bits). We assume the computation tasks are modeled as an i.i.d. Bernoulli process [5], i.e., at the beginning of each time slot, for each mobile device one computation task CT_i^t is requested with probability ρ where $0 < \rho < 1$. Denote $\zeta_i^t = 1$ if the ith mobile device gets computation task request at the tth time slot.

In our system, there exists no cache queue either in mobile devices or MEC servers. As a result, computation task CT_i^t can either be executed locally at the ith mobile device, or be

offloaded to the jth MEC server and then be executed, where j is the chosen MEC server for the ith mobile device by a system operation. Besides, if the energy of the ith mobile device is insufficient or $\zeta_i^t = 0$, the computation task at the tth time slot will be dropped, but the latter one will not be counted into the ratio of dropped tasks. Similar to [5], denote the computation mode indicators for the ith mobile device at the tth time slot as $I_{i,c}^t \in \{0,1\}$ with $c = \{l, r, d\}$, where $I_{i,l}^t = 1$, $I_{i,r}^t = 1$ and $I_{i,d}^t = 1$ indicate that the computation task is executed locally, executed remotely by a MEC server and dropped, respectively. Because there exists only three modes for the ith mobile device to choose at the tth time slot, those indicators should follow the equation as below:

$$I_{i,l}^t + I_{i,r}^t + I_{i,d}^t = 1, t \in \mathcal{T}, i \in \mathcal{N}. \quad (1)$$

C. Offloading Computation Model

Denote the indicator of the ith mobile device choosing the jth MEC server to offload as $c_{i,j}^t = 1$, where $c_{i,j}^t \in \{0,1\}$. Thus, the connection matrix $C^t \triangleq (c_{i,j}^t)_{N \times M}$ is time-varying, which is the same as the distance matrix. Similar to [5], each computation task is assigned to only one server when choosing the offloading computation mode, i.e.,

$$\sum_{j=1}^M c_{i,j}^t = 1, t \in \mathcal{T}, i \in \mathcal{N}, j \epsilon \mathcal{M}. \quad (2)$$

Denote the small-scale fading channel power gains from the ith mobile device to the jth MEC server by $\gamma_{i,j}^t$, which are assumed to be exponentially distributed with unit mean. Besides, each mobile device shares the same $\gamma_{i,j}^t$ at the tth time slot. According to communication theory, the channel power gain from the ith mobile device to the jth MEC server can be expressed by $h_{i,j}^t = \gamma_{i,j}^t g_0 (d_0/d_{i,j}^t)^\theta$, where d_0 denotes the reference distance, θ denotes the path-loss exponent and g_0 denotes the path-loss constant. As a result, we can obtain the achievable rate $\Gamma(h_{i,j}^t, p_i^t)$ by

$$\Gamma(h_{i,j}^t, p_i^t) = \omega \log_2(1 + h_{i,j}^t p_i^t / \sigma), t \in \mathcal{T}, i \in \mathcal{N}, j \epsilon \mathcal{M} \quad (3)$$

according to Shannon Theorem, where ω represents the bandwidth assigned to the ith mobile device, σ is the noise power at each MEC server and p_i^t is the transmit power who cannot exceed the maximum value $p^{\max}$. Note that we assume that every mobile device connecting to the jth MEC server is assigned the same bandwidth and each MEC server shares the same noise power, i.e., the bandwidth of each MEC server is equally divided into N sub-bands, and each mobile device is assigned to one sub-band with ω MHz in (3).

Denote the numbers of CPU cycles required to process one bit task by each mobile device and each MEC server as X_{mobile} and X_{server}, respectively. We assume that the computational abilities of MEC servers are constrained, i.e.,

$$\sum_{i \in \mathcal{N}} \mathrm{I}(I_{i,r}^t) \cdot c_{i,j}^t L X_{\mathrm{server}} \leq f_{\mathrm{server}}^{\max} \tau, t \in \mathcal{T}, j \epsilon \mathcal{M}, \quad (4)$$

where $f_{\mathrm{server}}^{\max}$ denotes the upper bound of each MEC server's CPU-cycle frequency and $\mathrm{I}(\cdot)$ is the indicator function.

Similar to [5], we take no consideration of execution latency consumed by the MEC server's execution process, i.e., for the

ith mobile device, the total execution latency of this mode equals to the transmission delay for the input task. Thus,

$$D_{i,\text{remote}}^t = \frac{L}{\Gamma\left(h_{i,j}^t, p_i^t\right)}, t \in \mathcal{T}, i \in \mathcal{N}, j\epsilon\mathcal{M}. \quad (5)$$

Meanwhile, we take no consideration of MEC server's energy consumption. Thus, the energy consumed by the ith mobile device can obtained by

$$E_{i,\text{remote}}^t = p_i^t \frac{L}{\Gamma\left(h_{i,j}^t, p_i^t\right)}, t \in \mathcal{T}, i \in \mathcal{N}, j\epsilon\mathcal{M}. \quad (6)$$

D. Local Computation Model

In order to execute L bits computation task successfully, LX_{mobile} CPU cycles are required. By applying the dynamic voltage and frequency scaling technologies (DVFS) [14], mobile devices can control the energy consumption and the execution latency. Thus, the total execution latency of this mode can be obtained by

$$D_{i,\text{local}}^t = \sum_{v=1}^{LX_{\text{mobile}}} (f_{i,v}^t)^{-1}, t \in \mathcal{T}, i \in \mathcal{N}. \quad (7)$$

According to [5], the energy consumed by the ith mobile device can obtained by

$$E_{i,\text{local}}^t = \sum_{v=1}^{LX_{\text{mobile}}} s(f_{i,v}^t)^2, t \in \mathcal{T}, i \in \mathcal{N}, \quad (8)$$

where s is the effective capacitance coefficient.

Moreover, we denote the upper bound of each mobile device's CPU-cycle frequency as $f_{\text{mobile}}^{\max}$, i.e., $\forall v \in \{1,2,\dots,LX_{\text{mobile}}\}$, $f_{i,v}^t \leq f_{\text{mobile}}^{\max}, t \in \mathcal{T}, i \in \mathcal{N}$.

E. Energy Harvesting Model

In order to embody the stochastic and intermitted nature of the renewable energy process [15], we assume that the harvestable energy $E_{i,H}^t$ of the ith mobile device at the beginning of the tth time slot is uniformly distributed with the maximum value of $E_H^{\max}$.

Similar to [5], the system needs to decide the amount of energy which will be stored in the battery of the ith mobile device. Denote this part of energy as e_i^t, then we have

$$0 \leq e_i^t \leq E_{i,H}^t, t \in \mathcal{T}, i \in \mathcal{N}. \quad (9)$$

We assume that other kinds of energy consumption besides local-execution and remote-execution is sufficient small. We use $\mathcal{E}(I_i^t, f_i^t, p_i^t)$ to denote the energy consumed by the ith mobile device at the tth time slot, where $I_i^t \triangleq [I_{i,l}^t, I_{i,r}^t, I_{i,d}^t]$ and $f_i^t \triangleq [f_{i,1}^t, \dots, f_{i,LX_{\text{mobile}}}^t]$. Then we can obtain $\mathcal{E}(I_i^t, f_i^t, p_i^t)$ by the following equation:

$$\mathcal{E}(I_i^t, f_i^t, p_i^t) = I_{i,l}^t E_{i,\text{local}}^t + I_{i,r}^t E_{i,\text{remote}}^t, t \in \mathcal{T}, i \in \mathcal{N}. \quad (10)$$

Denote the battery level of the ith mobile device at the tth time slot as b_i^t. Obviously, the energy consumption at each time slot cannot surpass the battery level, i.e.,

$$\mathcal{E}(I_i^t, f_i^t, p_i^t) \leq b_i^t, t \in \mathcal{T}, i \in \mathcal{N}. \quad (11)$$

Besides, b_i^t evolves according to the following equation:

$$b_i^{t+1} = b_i^t - \mathcal{E}(I_i^t, f_i^t, p_i^t) + e_i^t, t \in \mathcal{T}, i \in \mathcal{N}. \quad (12)$$

F. QoE-Cost Function

User's QoE [16] consists of execution delay and the penalty for dropping the task. Denote QoE-cost of the ith mobile device at the tth time slot as cost_i^t, then we can obtain the following equation:

$$\text{cost}_{\text{sum}}^t \triangleq \sum_{i\epsilon\mathcal{N}} \text{cost}_i^t$$
$$= \sum_{i\epsilon\mathcal{N}} \left[\mathcal{D}(I_i^t, f_i^t, p_i^t) + \emptyset \cdot \text{I}\left(\zeta_i^t \cap I_{i,d}^t\right)\right], t \in \mathcal{T}, i \in \mathcal{N}, \quad (13)$$

Where $\text{cost}_{\text{sum}}^t$ denotes the total QoE-cost of all mobile devices, $\emptyset$ denotes the weight of task dropping cost, and $\mathcal{D}(I_i^t, f_i^t, p_i^t)$ denotes the execution delay of the ith mobile device at the tth time slot, which is given by

$$\mathcal{D}(I_i^t, f_i^t, p_i^t) = \text{I}(\zeta_i^t = 1) \cdot$$
$$(I_{i,l}^t D_{i,\text{local}}^t + I_{i,r}^t D_{i,\text{remote}}^t), t \in \mathcal{T}, i \in \mathcal{N}. \quad (14)$$

G. Optimization Model

In our system, each MEC server has more computing power than each mobile device, i.e., when the cost of offloading the task is still lower than the cost of local execution, the more computation tasks executed remotely, the better user's quality of experience. We can obtain the number of offloading computation tasks at tth time slot by

$$\sum_{i\epsilon\mathcal{N}} \text{I}\left(\zeta_i^t \cap I_{i,r}^t\right), t \in \mathcal{T}, \quad (15)$$

where $\text{I}(\cdot)$ is the indicator function. As a result, we focus on two optimization goals, i.e., the average weighted sum QoE-cost minimization and the number of offloading computation tasks maximization.

Denote the system operation at the tth time slot as

$$SO^t \triangleq [I^t, f^t, p^t, C^t, e^t], t \in \mathcal{T}, \quad (16)$$

in which $I^t \triangleq [I_{1,c}^t, \dots, I_{N,c}^t]$, $f^t \triangleq [f_1^t, \dots, f_N^t]$, and $p^t \triangleq [p_1^t, \dots, p_N^t]$, $e^t \triangleq [e_1^t, \dots, e_N^t]$. Consequently, the optimization problem can be expressed as

$$\mathcal{P}_1: \min_{SO^t} \lim_{T\to\infty} \frac{1}{T}\sum_{t=0}^{T-1} \text{cost}_{\text{sum}}^t - \psi \cdot \sum_{i\epsilon\mathcal{N}} \text{I}\left(\zeta_i^t \cap I_{i,r}^t\right)$$

$$s.t. \ (1), (2), (4), (9), (10), (11)$$

$$0 \leq f_{i,v}^t \leq f_{\text{mobile}}^{\max}, t \in \mathcal{T}, i \in \mathcal{N}, v \in \{1, \dots, LX_{\text{mobile}}\} \quad (17)$$

$$\mathcal{E}(I_i^t, f_i^t, p_i^t) \leq E_{\max}, t \in \mathcal{T}, i \in \mathcal{N} \quad (18)$$

$$0 \leq p_i^t \leq p^{\max}, t \in \mathcal{T}, i \in \mathcal{N} \quad (19)$$

$$I_{i,c}^t \in \{0,1\}, c \in \{l,r,d\}, t \in \mathcal{T}, \quad (20)$$

where ψ defines the weight of the second optimization goal. (17) and (19) incarnate the constraints of mobile devices' CPU-cycle frequency and maximum transmit power, respectively. (18) incarnates the upper bound of battery discharging for security reasons, i.e., the amount of energy output energy cannot exceed $E_{\max}$ at each time slot. (20) represents the 0-1 indicator constraint which has been described in subsection B of Section III.

IV. A Low-Complexity Algorithm for Joint QoE-Cost and Cell-Capacity Optimization

In this section, we will develop the LODCO-Based Genetic Algorithm to solve $\mathcal{P}_1$ on account of the LODCO algorithm [5]. First of all, we will convert the original problem which is time-dependent to a per-time slot deterministic problem $\mathcal{P}_2$ by taking advantages of Lyapunov Optimization. Then the LODCO Algorithm will be upgraded and reconstructed for the multi-user and multi-server MEC system by virtue of Genetic Algorithm and Greedy Strategy. We will not demonstrate the details of the LODCO Algorithm, but we will explain closely about why we can apply it to our model.

A. Drift Plus Penalty Formula

Lyapunov optimization demands that the allowable action sets are i.i.d., which cannot be satisfied by the time-dependent battery queues of mobile devices modeled in this paper. Similar to one part of the LODCO Algorithm, for each mobile device we use the perturbation parameter θ (which is lower bounded by $\tilde{E}_{\max} + V\emptyset$) to define virtual battery queue $\tilde{b}_i^t$ by

$$\tilde{b}_i^t \triangleq b_i^t - \theta, t \in \mathcal{T}, i \in \mathcal{N}, \tag{21}$$

where $\tilde{E}_{\max} \triangleq \min\{\max\{sLX_{\text{mobile}}(f_{\text{mobile}}^{\max})^2, p^{\max}\tau\}, E_{\max}\}$.

Besides, if a task requested at the tth time slot is being executed locally, the optimal frequencies of the LX_{mobile} CPU cycles should be the same, i.e., $f_{i,v}^t = f_i^t, i \in \{1, \dots, LX_{\text{mobile}}\}$, which can be obtained by *Inequality of arithmetic and geometric means.*

According to the analysis above, we define the Lyapunov function as

$$L(t) \triangleq \frac{1}{2}\sum_{i\in\mathcal{N}}\left(\tilde{b}_i^t\right)^2 = \frac{1}{2}\sum_{i\in\mathcal{N}}(b_i^t - \theta)^2, t \in \mathcal{T}. \tag{22}$$

Thus, the conditional Lyapunov drift can be written as

$$\Delta(t) \triangleq \mathbb{E}\left[L(t+1) - L(t)\big|\tilde{b}^t\right], t \in \mathcal{T}, \tag{23}$$

where $\tilde{b}^t \triangleq [\tilde{b}_1^t, \dots, \tilde{b}_N^t]$. Then the Lyapunov drift-plus-penalty function can be written as

$$\Delta_V(t) \triangleq \Delta(t) + V \cdot \mathbb{E}\left[\text{cost}_{\text{sum}}^t\big|\tilde{b}^t\right], t \in \mathcal{T}. \tag{24}$$

Because of the energy evolution equation (12), we can obtain that

$$\left(\tilde{b}_{i+1}^t\right)^2 \leq \left(\tilde{b}_i^t\right)^2 +$$
$$2\tilde{b}_i^t\left(e_i^t - \mathcal{E}(I_i^t, f_i^t, p_i^t)\right) + (e_i^t)^2 + \mathcal{E}^2(I_i^t, f_i^t, p_i^t), t \in \mathcal{T}, i \in \mathcal{N}. \tag{25}$$

Denotes the upper bound of the real energy consumed by the ith mobile device at the t th time slot as $\tilde{E}_{\max}$, we have $\mathcal{E}(I_i^t, f_i^t, p_i^t) \leq \tilde{E}_{\max}$. Because $e_i^t \leq E_{i,H}^t$ and $E_{i,H}^t$ are i.i.d. among different time slots with the maximum value of $E_H^{\max}$ for each mobile device, we can get $e_i^t \leq E_H^{\max}$. As a result, we have

$$\Delta(t) \leq \sum_{i\in\mathcal{N}}\mathbb{E}\left[\tilde{b}_i^t\left(e_i^t - \mathcal{E}(I_i^t, f_i^t, p_i^t)\right)\big|\tilde{b}^t\right] + C, t \in \mathcal{T}, \tag{26}$$

where $C = \frac{N}{2}(\tilde{E}_{\max} + E_H^{\max})^2$. Combing with (24), we can obtain that

$$\Delta_V(t) \leq \sum_{i\in\mathcal{N}}\tilde{b}_i^t\left(e_i^t - \mathcal{E}(I_i^t, f_i^t, p_i^t)\right) +$$
$$V \cdot \mathbb{E}\left[\text{cost}_{\text{sum}}^t\big|\tilde{b}^t\right] + C, t \in \mathcal{T}, \tag{27}$$

which means $\Delta_V(t)$ is upper bounded.

Obviously, the upper bound of $\Delta_V(t)$ has the same structure with the problem defined in [5]. Thus, the LODCO Algorithm can hopefully be applied to decide $\boldsymbol{SO}^t$ by solving the following deterministic problem

$$\mathcal{P}_2: \min_{\boldsymbol{SO}^t} \sum_{i\in\mathcal{N}}\tilde{b}_i^t\left(e_i^t - \mathcal{E}(I_i^t, f_i^t, p_i^t)\right) + V \cdot$$
$$\mathbb{E}\left[\text{cost}_{\text{sum}}^t\big|\tilde{b}^t\right] \in \mathcal{T}, t \in \mathcal{T}, i \in \mathcal{N},$$

which subjects to every constraint condition of $\mathcal{P}_1$. In the above deterministic problem $\mathcal{P}_2$, we ignore the second optimal goal because it is not relative to the stability and reliability of the system. Besides, it can be added to $\mathcal{P}_2$ directly. We will describe the process in detail in subsection C of this section.

B. Application of the LODCO Algorithm

$\mathcal{P}_2$ is an optimized format of $\mathcal{P}_1$ by Lyapunov Optimization except maximizing the number of offloading computation tasks. According to the LODCO Algorithm, the problem can be decomposed to two sub problems. The first one is to find the optimal energy harvesting, and the second one is to decide the optimal computation modes.

Optimal energy harvesting: the optimal amount of harvested energy e_i^{t*} for the ith mobile device can be obtained by solving the following problem:

$$\min_{0 \leq e_i^t \leq E_{i,H}^t, i\in\mathcal{N}, t\in T} \sum_{i\in\mathcal{N}}\tilde{b}_i^t e_i^t.$$

Because each mobile device's decision on optimal energy harvesting is mutually independent, the optimal e_i^{t*} can be obtained separately for each mobile device. Thus, we obtain e_i^{t*} for each mobile device by (21) in [5].

Decide the computation modes: we need to obtain the mode with the minimum value of $J_{CO}(I_i^t, \boldsymbol{f}_i^t, p_i^t)$ for each mobile device, where

$$J_{CO}(I_i^t, \boldsymbol{f}_i^t, p_i^t) \triangleq I(I_{i,l}^t) \cdot J_{\text{mobile}}^t(f_i^t) +$$
$$I(I_{i,r}^t) \cdot J_{\text{server}}^t(p_i^t) + I(I_{i,d}^t) \cdot V\emptyset, t \in \mathcal{T}, i \in \mathcal{N}, \tag{28}$$

where $J_{\text{mobile}}^t(f_i^t)$ and $J_{\text{server}}^t(p_i^t)$ denote the optimal goals for $\mathcal{P}_{\text{ME}}$ and $\mathcal{P}_{\text{SE}}$ [2], which are sub-problems for local-execution mode and remote-execution mode, respectively.

We have the same Lyapunov structure with [5], thus we can obtain the optimal $\boldsymbol{SO}^t$ when each mobile device can make decision separately. However, there exists correlation between any two mobile devices when choosing the computation modes,

---

[2] Both $\mathcal{P}_{\text{ME}}$ and $\mathcal{P}_{\text{SE}}$ are the sub-problems defined in [5], where the former one is a problem to obtain the optimal CPU-cycle frequency for a task being executed locally and the latter one is a

problem to obtain the optima transmit power for computation offloading. Both of them are parts of the LODCO Algorithm.

especially when choosing the offloading computation mode. The reason is that our system has more than one server. As a result, the LODCO Algorithm cannot be utilized directly on our problem. In order to solve the problem, we propose the LODCO-Based Genetic Algorithm with Greedy Policy for the multi-user and multi-server system.

At the tth time slot, the genetic algorithm will be used to obtain the minimum value of $\mathcal{P}_2$ when each mobile device's battery energy level is sufficient for local execution or offloaded execution. Otherwise, the key-value pair method will be used by virtue of ϵ-greedy policy [17].

C. The LODCO-Based Genetic Algorithm with Greedy Policy

In this section, we demonstrate the details of proposed algorithm.

Determine the decision variables. There will be much redundancy if we use the system operation $\boldsymbol{SO}^t$ as the decision variables directly, which can lead to slow convergence of genetic algorithm. To overcome the above problem, we define the decision vector $\boldsymbol{x}^t$ for the whole system as

$$\boldsymbol{x}^t \triangleq [\boldsymbol{x}_1^t, \boldsymbol{x}_2^t, \dots, \boldsymbol{x}_N^t], t \in \mathcal{T}, \qquad (29)$$

where

$$\boldsymbol{x}_i^t \triangleq [I_{i,l}^t, I_{i,d}^t, c_{i,1}^t, c_{i,2}^t, \dots, c_{i,M}^t], t \in \mathcal{T}, i \in \mathcal{N} \qquad (30)$$

is the decision vector for the ith mobile device. Denote the length of $\boldsymbol{x}^t$ as $|\boldsymbol{x}^t|$, then we can obtain that $|\boldsymbol{x}^t|$ is $N \times (M + 2)$. Besides, each element in $\boldsymbol{x}^t$ is either 0 or 1, i.e.,

$$x_k^t \in \{0,1\}, t \in \mathcal{T}, k \epsilon \{1,2,\dots,|\boldsymbol{x}^t|\}. \qquad (31)$$

Compared with system operation $\boldsymbol{SO}^t$, $\boldsymbol{I}^t$ is decomposed to $I_{i,l}^t$ and $I_{i,d}^t$ in $\boldsymbol{x}^t$, and then $I_{i,r}^t$ is replaced to $[c_{i,1}^t, c_{i,2}^t, \dots, c_{i,M}^t]$.. We ignore $\boldsymbol{f}^t$, $\boldsymbol{p}^t$ and $\boldsymbol{e}^t$ in $\boldsymbol{SO}^t$ because all of them can be obtained by the LODCO Algorithm separately. For convenience, we use the index between 1 and $N \times (M + 2)$ of $\boldsymbol{x}^t$ to describe each element of it. For example, $I_{i,l}^t$ can be written as $x_{1+(i-1)\times(M+2)}^t$, $c_{i,M}^t$ can be written as $x_{i\times(M+2)}^t$ and so on.

Modify the constraints. The computational abilities of MEC servers are constrained in (4), which can be written as

$$S_{\text{UB}} \cdot LX_{\text{server}} \leq f_{\text{server}}^{\max}\tau, t \in \mathcal{T}, \qquad (32)$$

where S_{UB} is the maximum number of mobile devices that could connect to a MEC server simultaneously. Having considered the fact that S_{UB} must be an integer, we can obtain it by

$$S_{\text{UB}} \leq \lfloor f_{\text{server}}^{\max}\tau/LX_{\text{server}}\rfloor. \qquad (33)$$

Thus, we can rewrite (4) in the following way:

$$\textstyle\sum_{i=1}^N x_{k+(i-1)\times(M+1)}^t \leq S_{\text{UB}}, t \in \mathcal{T}, k\epsilon\{3,4,\dots,M+2\}. \quad (34)$$

By combining (1) with (2), we have the following constraint:

$$\textstyle\sum_{k=1}^{M+2} x_{k+(i-1)\times(M+1)}^t \leq 1, t \in \mathcal{T}, i \in \mathcal{N}, \qquad (35)$$

which means each mobile device can choose only one mode.

Modify the optimization goals. We formulate $\mathcal{P}_3$ by adding the second optimal goal to $\mathcal{P}_2$, i.e.,

$$\mathcal{P}_3: \min_{\boldsymbol{x}^t} \sum_{i=1}^N \tilde{b}_i^t \left(e_i^t - x_{1+(i-1)\times(M+1)}^t \cdot E_{i,\text{local}}^{t*} - \sum_{k=3}^{M+2} x_{k+(i-1)\times(M+1)}^t \cdot E_{i,j}^{t*} \right)$$
$$+ \sum_{i=1}^N \left(x_{1+(i-1)\times(M+1)}^t \cdot D_{i,\text{local}}^{t*} + \sum_{k=3}^{M+2} x_{k+(i-1)\times(M+1)}^t \cdot D_{i,j}^{t*} + \emptyset \cdot x_{2+(i-1)\times(M+1)}^t \right)$$
$$+ \psi \cdot \sum_{i=1}^N \sum_{k=3}^{M+2} x_{k+(i-1)\times(M+1)}^t, t \in \mathcal{T},$$

where $E_{i,\text{local}}^{t*}$ and $D_{i,\text{local}}^{t*}$ are the optimal energy consumption and execution delay of local execution for the ith mobile device separately, $E_{i,j}^{t*}$ and $D_{i,j}^{t*}$ are the optimal energy consumption and execution delay when the ith mobile device offloads its tasks to the jth MEC server, respectively.

According to the LODCO Algorithm, at the very beginning of the iteration, the computation task has to be dropped because of the insufficient battery energy, which means there exists no optimal energy consumption or execution delay, i.e., $\mathcal{P}_3$ will not work. Besides, if the sub-problems, i.e., $\mathcal{P}_{\text{ME}}$ and $\mathcal{P}_{\text{SE}}$, are infeasible, $\mathcal{P}_3$ will not work, too. In this case, we develop a key-value pair method by virtue of ϵ-greedy policy. To be more specific, we use a map to store each mobile device and its optimal MEC server which is chosen to offload. At each time slot, the key-value pair "i-j" with the minimum $J_{\text{server}}^t(p_i^t)$ will be chosen to compare with other modes for the ith mobile device. We will always choose the offloading execution mode

whatever $J_{\text{server}}^t(p_i^t)$ is the minimum among three modes with the probability of ϵ. With the probability of $1 - \epsilon$, we will choose the mode with the minimum value of optimal goal among $J_{\text{mobile}}^t(f_i^t)$, $J_{\text{server}}^t(p_i^t)$ and $V\emptyset$, where $J_{\text{server}}^t(p_i^t)$ and $J_{\text{mobile}}^t(f_i^t)$ are the optimization goals for $\mathcal{P}_{\text{SE}}$ and $\mathcal{P}_{\text{ME}}$, respectively.

Algorithm 1
LODCO-Based Genetic Algorithm with Greedy Policy
1: At the beginning of the tth time slot, initialize flag[M] with 0 and establish a map to store the indexes of mobile device and the corresponding chosen MEC server.
2: Initialize Boolean variable *useKeyValuePair* with **false**.
3: **for** each mobile device i **do**
4: Obtain the task request indicator ζ_i^t, virtual energy queue $\tilde{b}_i^t$ and harvestable energy $E_{i,H}^t$.
5: Generate the location of each mobile device.

6:　　Obtain the optimal harvested energy e_i^{t*} by the LODCO Algorithm.

7:　　Obtain the optimal f_i^{t*} for local execution by the LODCO Algorithm, then record the optimal value $J_{\text{mobile}}^t(f_i^t)$. If the battery energy is insufficient or $\mathcal{P}_{\text{ME}}$ is infeasible, set *useKeyValuePair* as **true**.

8:　　**for** each MEC server j **do**

9:　　　Obtain the channel power gain $h_{i,j}^t$ from the ith mobile device to the jth MEC server by $h_{i,j}^t = \gamma_{i,j}^t g_0(d_0/d_{i,j}^t)^\theta$.

10:　　　Obtain the optimal $p_{i,j}^{t*}$ from the ith mobile device to the jth MEC server by the LODCO Algorithm, then record the optimal value $J_{\text{server}}^t(p_{i,j}^t)$. If the battery energy is insufficient or $\mathcal{P}_{\text{SE}}$ is infeasible, set *useKeyValuePair* as **true**.

11:　　　Choose the optimal p_i^{t*} by selecting the one with minimum $J_{\text{server}}^t(p_{i,j}^t)$, denote as $J_{\text{server}}^t(p_i^t)$ and then record j.

12:　　**end for**

13:　　Compare $J_{\text{mobile}}^t(f_i^t)$, $J_{\text{server}}^t(p_i^t)$ and $V\emptyset$, choose the mode with the minimum value and set the corresponding indicator variable $I_{i,c}^t$ as 1.

14:　　**if** $I_{i,r}^t = 1$ **then**

15:　　　obtain the ith mobile device and the corresponding jth MEC server, then insert them into the map with key i and value j.

16:　　**end if**

17: **end for**

18: **if** *useKeyValuePair* $==$ **false then**

19:　　Use Genetic Algorithm to solve $\mathcal{P}_3$ with constraints of (31), (34), (35).

20: **else**

21:　　Call the Key-value Pair Method.

22: **end if**

23: Obtain each mobile device's execution cost and energy consumption at each time slot, then calculate the proportion of each mode.

24: Update the battery level for each mobile device.

25: Update t to $t+1$.

Algorithm 1 gives the main parts of the proposed algorithm, then we will show the details of Key-value Pair Method.

Subroutine 1 Key-value Pair Method
1:　**while** the map is not **null do**
2:　　Obtain the key-value pair "i-j" with the minimum $J_{\text{server}}^t(p_i^t)$.
3:　　**if** rand() $< \epsilon$ **then**
4:　　　**if** flag[j] $\leq S_{\text{UB}}$ **then**
5:　　　　Remove the key-value pair "i-j" from the map and then flag[j]++ no matter whether $J_{\text{server}}^t(p_i^t)$ is the minimum among $J_{\text{mobile}}^t(f_i^t)$, $J_{\text{server}}^t(p_i^t)$ and $V\emptyset$. Then set $J_{\text{server}}^t(p_{i,j}^t)$ as **inf**.
6:　　　**else**

7:　　　　**if** $\min\{J_{\text{server}}^t(p_{i,:}^t)\}$!= **inf then**[3]
8:　　　　　Find the optimal j by $\min\{J_{\text{server}}^t(p_{i,:}^t)\}$ and the insert them to the map. Then **continue**.
9:　　　　**else**
10:　　　　　Select the optimal mode from other 2 modes: local execution and dropping the task. Then remove the corresponding key-value pair from map.
11:　　　　**end if**
12:　　　**end if**
13:　　**else if** rand()$\geq \epsilon$ **then**
14:　　　Compare $J_{\text{mobile}}^t(f_i^t)$, $J_{\text{server}}^t(p_i^t)$ and $V\emptyset$, choose the mode with the minimum value and set the corresponding indicator variable $I_{i,c}^t$ as 1 (we have to do the search again because it is possible that $J_{\text{server}}^t(p_i^t)$ has been modified).
15:　　　**if** $I_{i,r}^t = 1$ **then**
16:　　　　**if** flag[j] $\leq S_{\text{UB}}$ **then**
17:　　　　　Remove the key-value pair "i-j" from the map and flag[j]++. Then set $J_{\text{server}}^t(p_{i,j}^t)$ as **inf**.
18:　　　　**else**
19:　　　　　**if** $\min\{J_{\text{server}}^t(p_{i,:}^t)\}$!= **inf then**
20:　　　　　　Find the optimal j by $\min\{J_{\text{server}}^t(p_{i,:}^t)\}$ and then insert them to the map. Then **continue**.
21:　　　　　**else**
22:　　　　　　Select the optimal mode from other two modes: local execution and dropping the task. Then remove the corresponding key-value pair from map.
23:　　　　　**end if**
24:　　　　**end if**
25:　　　**else**
26:　　　　Keep the corresponding $I_{i,c}^t = 1$ without change and then remove the corresponding key-value pair from map.
27:　　　**end if**
28:　　**end if**
29: **end while**

V. SIMULATION RESULTS

In this section, we will demonstrate the results of the proposed algorithm and verify its effectiveness. Then, we will show the impact of the system parameters by control variable method. As mentioned in section IV, we will not elaborate upon the details about the verification of the LODCO Algorithm.

The simulation was run on a laptop with an Intel Core 2.5 GHz i7-4710MQ CPU. The algorithm was implemented in MATLAB R2015b and was given up to 8 GB of memory if needed.

In our system, the harvestable energy $E_{i,H}^t$ is uniformly distributed with the maximum value of $E_H^{\max}$, where $E_H^{\max}$ can be obtained by average EH power P_H, i.e.,

$$E_H^{\max} = P_H \cdot 2\tau, t \in \mathcal{T}. \tag{36}$$

---

[3] $J_{\text{server}}^t(p_{i,:}^t)$ is defined as $[J_{\text{server}}^t(p_{i,1}^t), J_{\text{server}}^t(p_{i,2}^t), \dots, J_{\text{server}}^t(p_{i,M}^t)]$.

We assume that $P_H = 12$ mW, $g_0 = -40$ dB (path-loss constant), $\varsigma = 10^{-28}$ (effective capacitance coefficient), $\tau = \emptyset = 2$ ms (time slot length and the weight of the task dropping cost, respectively). In addition, $\omega = 1$ MHz (bandwidth of one sub-band), $\sigma = 10^{-13}$ W (noise power at each MEC server), $f_{mobile}^{max} = f_{server}^{max} = 1.5$ GHz (upper bounds of each mobile device and each MEC server's CPU-cycle frequency, respectively), $E_{max} = 2$ mJ (the upper bound of the real energy consumed by mobile device at each time slot), $L = 1000$ bits (size of each computation task), $V = 10^{-5}$ (coefficient of the penalty in Lyapunov Optimization), and $X_{server} = X_{mobile} = 5900$ cycles per byte (the numbers of CPU cycles required to process one bit task by each mobile device and each MEC server). Besides, we assume there are 10 mobile devices and 5 MEC servers, i.e., $N = 10$, $M = 5$, and $E_{min} = 0.02$ mJ (the non-zero lower bound of mobile devices defined in [5]).

In the following part, the default value of ρ (task request probability), ψ (the weight of the second optimization goal) and ϵ in Algorithm 1 are 0.7, 0.002 and 0.1 unless stated, respectively. $\gamma_{i,j}^t$ (the small-scale fading channel power gains at the tth time slot) is exponentially distributed with mean 1. Besides, we assume the maximum distance between random mobile and random MEC server is 80 m unless stated, which is the upper bound for the uniform distribution.

Because no work has been found to solve the same problem as we have done, we introduce one benchmark algorithm, namely, the LODCO-Based Greedy Algorithm, it works as follows: At each time slot, this algorithm always chooses the mode with the minimum target value of $\mathcal{P}_2$ between local execution and offloaded execution for each mobile device, which means it will not take the second optimization goal into consideration; otherwise, if neither one is infeasible, then the computation tasks will be dropped. Besides, the greedy policy in this algorithm has the same structure with Key-value Pair method.

A. Validation of Effectiveness

In this subsection, we will compare the effectiveness of the LODCO-Based Genetic Algorithm with Greedy Policy compared with the LODCO Algorithm and the LODCO-Based Greedy Algorithm. As shown in Fig. 2, the battery energy level of each mobile device demonstrates the feasibility of our improvement on the LODCO Algorithm. The energy level of each mobile device keeps accumulating at earlier stage, and finally stabilizes around the perturbed energy level, which exactly keeps the advantages of the LODCO Algorithm. Under current parameters, the battery energy level of each mobile device stabilizes at the beginning of the 200th time slot.

Fig. 3 demonstrates the average QoE-cost of all mobile devices at each time slot obtained by the LODCO-Based Genetic Algorithm with Greedy Policy. As we can see, the average QoE-cost declines steeply at the early stage and then stabilizes at a low level, which is the embodiment of inheriting advantages from the LODCO Algorithm.
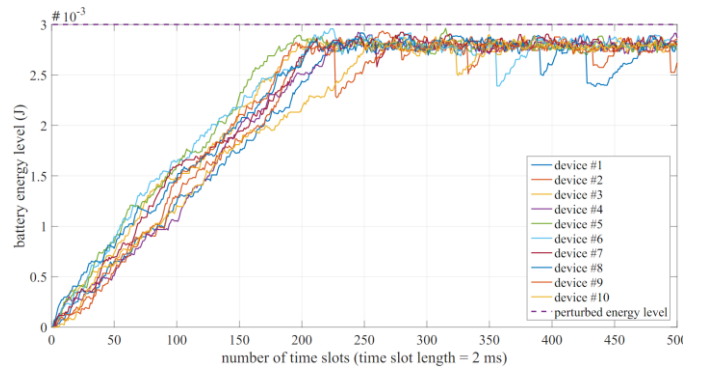

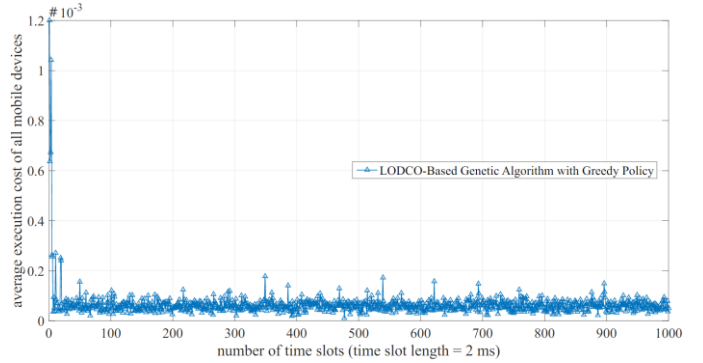Fig. 2. Battery energy level of each mobile device vs. time.

Fig. 3. Average QoE-Cost of all mobile device vs. time.

As depicted in Fig. 4, the Y-axis describes the average value of each mobile device's energy level during 500 time slots. As can be seen, each energy level is confined within $[0.00002, 0.005]$ J[4], which conforms to the theoretical results derived in the LODCO Algorithm.
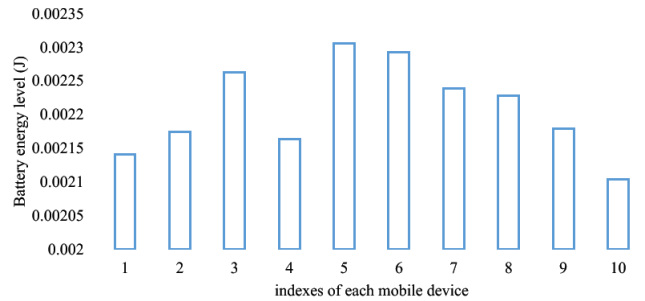

Fig. 4. Average energy level of each mobile device.

Fig. 5 demonstrates the ratio of each chosen modes in our multi-user and multi-server system by the LODCO-Based Genetic Algorithm with Greedy Policy. At the very earlier stage, plenty of computation tasks are dropped due to the insufficient energy level. During the first 150 time slots, the average ratio of dropped tasks is 1.8942%, which significant decreases to 0 along with the ascending battery energy level of each mobile device. Meanwhile, the ratio of offloading tasks (which is 94.6291%) clearly greater than the ratio of locally-executed tasks (which is 3.4767%) and the average ratio of offloading tasks obtained by the LODCO Algorithm (which is 84.4401%),

---

[4] 0.005 and 0.00002 are the specific value of $\theta + E_H^{max}$ and E_{min}, respectively.

which means the LODCO-Based Genetic Algorithm with Greedy Policy can obtain better performance than the LODCO Algorithm can do in terms of maximizing the number of offloading computation tasks.
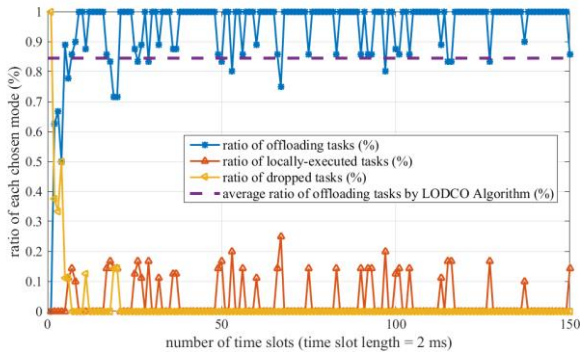

Fig. 5. The ratio of each chosen modes vs. time.

In Fig. 6, the performance of the LODCO-Based Genetic Algorithm with Greedy Policy is compared with that of the benchmark algorithm (the LODCO-Based Greedy Algorithm) on the second optimization goal (the number of offloading computation tasks maximum).

As depicted in Fig. 6, the average ratio of offloading tasks obtained by the LODCO-Based Genetic Algorithm with Greedy Policy (which is 95.0698%) is greater than the one obtained by the LODCO-Based Greedy Algorithm (which is 92.8549%[5]) , while both ratios are larger than the average ratio of offloading tasks obtained by the LODCO Algorithm (which is 84.4401%) . The results prove that our algorithm can obtain better cell capacity than the benchmark algorithm can do.
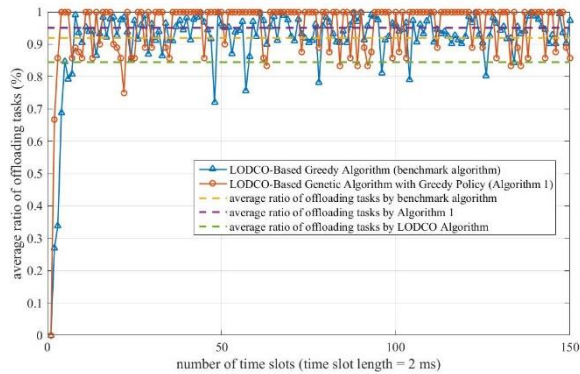

Fig. 6. Average ratio of offloading tasks by different algorithms.

B. Effects of system parameters

In this subsection, we will demonstrate the impacts of system parameters on the performance of the proposed algorithm.

Fig. 7 depicts the impact of the maximum distance between random mobile device and random MEC server on cell capacity. As can be seen, along with the increase of the maximum distance, the average ratio of offloading tasks gradually decreases. When the distance is arbitrarily far, the number of offloading

---

[5] Considering that this is a stochastic optimization problem, the simulation results may exist slightly difference.

computation tasks will be zero. The reason is that the channel power gain grows with the distance between each mobile device and each MEC server, which will lead to larger energy consumption and longer execution delay. As a result, more and more mobile devices will choose to execute the computation tasks locally.
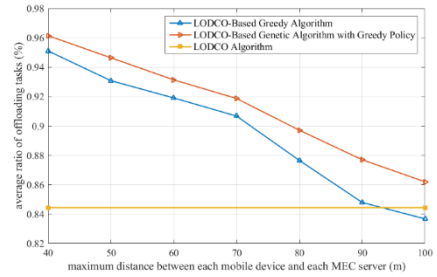

Fig. 7. Average ratio of offloading tasks vs. maximum distance.

As depicted in Fig. 8, along with the increase of ϵ, which belong to $[0,1]$, the average ratio of offloading tasks gradually increases with a slowdown rate and finally converges to the specific value (which is 97.5731%). The reason is that the LODCO-Based Genetic Algorithm with Greedy Policy is based on the ϵ-Greedy Strategy, i.e., a greater ϵ will bring a lager probability to choose the offloading mode.
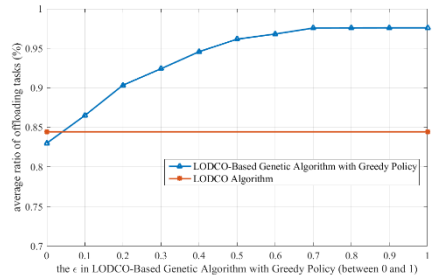

Fig. 8. Average ratio of offloading tasks vs. ϵ.

As depicted in Fig. 9, along with the increase of ψ, the average ratio of offloading tasks gradually increases with a slowdown rate and finally converges to the specific value (which is 98.5315%). The reason is that ψ determines the weight of the second optimization goal, which is an important component of $\mathcal{P}_3$.
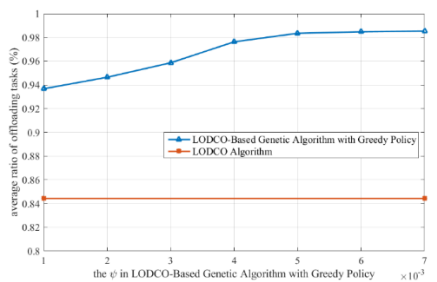

Fig. 9. Average ratio of offloading tasks vs. ψ.

VI. Conclusions

In this paper, we investigate a multi-user and multi-server mobile-edge computing system with energy harvesting devices. Then we propose one algorithm to obtain the lowest execution cost and largest number of offloading computation tasks based on the LODCO Algorithm, i.e., the LODCO-Based Genetic Algorithm with Greedy Policy, which is an online algorithm with low-complexity. Most importantly, it has no need of too much priori knowledge. After simulation and performance analysis, we can see that the proposed algorithm inherits every advantage from the LODCO Algorithm and adapts to the more complex environment perfectly and offers more than 10% ratio of offloading computation tasks. The proposed algorithm can choose the offloading mode as far as possible, which can bring resource-limited MEC servers' superiority into full play. In conclusion, our study provides a viable strategy to design a complex system which much more approaches to reality.

References

[1] Y.C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing – A key technology towards 5G", ETSI White Paper, 2015.

[2] T.X. Tran, H. Abolfazl, P. Pandey, and D. Pompili, "Collaborative mobile edge computing in 5G networks: New paradigms, scenarios, and challenges", IEEE Communication Magazine, vol. 55, 2017, pp. 54-61.

[3] S. Barbarossa, S. Sardellitti, and P.D. Lorenzo, "Communicating while computing: Distributed mobile cloud computing over 5G heterogeneous networks", IEEE Single Processing Magazine, vol. 31, 2014, pp. 45-55.

[4] S. Sudevalayam and P. Kulkarni, "Energy harvesting sensor nodes: Survey and implications", IEEE Communications Surveys & Tutorials, vol. 13, 2011, pp. 443-461.

[5] Y.Y. Mao, J. Zhang, K. B. Letaief, "Dynamic computation offloading for mobile-edge computing with energy harvesting devices", IEEE Journal of Selected Areas Communications, vol. 34, 2016, pp. 3590-3605.

[6] M.H. Min, D.J. Xu, L. Xiao, Y.L. Tang, and D. Wu, "Learning-based computation offloading for IoT devices with energy harvesting", arXiv: 1712.08768, 2017, unpublished.

[7] X.F. Chen, H.G. Zhang, C. Wu, S.W. Mao, Y.S. Ji, and M. Bennis, "Performance optimization in mobile-edge computing via deep reinforcement learning", arXiv: 1804.00514, 2018, unpublished.

[8] X.H. Ge, S. Tu, G.Q. Mao, C.X. Wang, and T. Han, "5G ultra-dense celluar networks", IEEE Wireless Communications, vol. 23, 2016, pp. 71-79.

[9] A. Alfayly and L.F. Sun, "QoE-driven LTE downlink scheduling for multimedia services", Ph.D. thesis, Plymouth University, 2016.

[10] C.F. Liu, M. Bennis, and H.V. Poor, "Latency and reliability-aware task offloading and resource allocation for mobile edge computing", IEEE Global Communications Conference Workshops (GLOBECOM Workshops), Singapore, 2017.

[11] M. Masoudi, B. khamidehi, and C. Cavdar, "Green cloud computing for multi cell networks", IEEE Wireless Communications and Networking Conference (WCNC), San Francisco, USA, 2017.

[12] T.X. Tran and D. Pompili, "Joint task offloading and resource allocation for multi-server mobile-edge computing networks", arXiv: 1705.00704, 2017, unpublished.

[13] L. Yang, J.N. Cao, Z.Y. Wang, and W.G. Wu, "Network aware multi-user computation partitioning in mobile edge clouds", International Conference on Parallel Processing (ICPP), Bristol, United Kingdom, 2017.

[14] Y.Y. Mao, C.S. You, J. Zhang, K.B. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective", IEEE Communication Surveys & Tutorials, vol. 19, 2017, pp. 2322-2358.

[15] L.B. Huang and M.J. Neely, "Utility optimal scheduling in energy-harvesting networks", IEEE/ACM Transactions on Networking, vol. 21, 2013, pp. 1117-1130.

[16] W.W. Zhang, Y.G. Wen, K. Guan, D. Kilper, H.Y. Luo, and D.O. Wu, "Energy-optimal mobile cloud computing under stochastic wireless channel", IEEE Transactions on Wireless Communications, vol. 12, 2013, pp. 4569-4581.

[17] S. Bubeck and N. Cesa-Bianchi, "Regret analysis of stochastic and nonstochastic multi-armed bandit problems", Foundations and Trends® in Machine Learning, vol. 5, 2012, pp. 1-22.