Dynamic Computation Offloading for MIMO Mobile Edge Computing Systems With Energy Harvesting

Wen Zhou, Ling Xing, *Member, IEEE*, Junjuan Xia, Lisheng Fan, and Arumugam Nallanathan, *Fellow, IEEE*

***Abstract*—By providing spatial diversity gain, the incorporation of multiple antennas into mobile edge computing (MEC) systems can improve the transmission performance. Meanwhile, employing energy harvesting (EH) helps enhance the system sustainability. In this paper, we focus on multi-input multi-output (MIMO) MEC systems with EH and studies the computation offloading. The design objective is to minimize the time average of a weighted sum of energy consumption and execution delay, meanwhile stabilizing the battery energy queue. To this end, we formulate the problem as a statistic program and propose a dynamic computation offloading (DCO) algorithm in which the transmitter covariance matrix, CPU-cycle frequencies for local computing, and partial offloading ratio are jointly optimized. Based on Lyapunov optimization, the program is first transformed into a nonconvex per-time slot problem. Then, we solve it by the successive convex approximation (SCA) technique, where a sequence of convex problems are created and solved. Simulation results demonstrate that the proposed algorithm is asymptotically optimal and outperforms several benchmark schemes in terms of both the average system cost and task drop ratio.**

***Index Terms*—Mobile edge computing, MIMO, Lyapunov optimization, energy harvesting.**

I. Introduction

Increasing mobile terminals and diversified service types bring challenges to mobile networks. To address it, mobile edge computing (MEC) is proposed to partly transfer the functions originally located in the cloud computing center to the edge of the network [1]. In this way, network congestion and system pressure can be relieved. On the other front, mobile terminals, especially small Internet of Things (IoT) devices, are often plagued by limited battery energy [2]. Fortunately, energy harvesting (EH) technology can improve this situation by capturing green energy. The integration of EH into MEC will enhance sustained computation ability of mobile devices, having potential applications such as wearable medical systems, environmental monitoring, and disaster relief, which has received a lot of attention [3]–[12].

The MEC systems with EH, termed as MEC-EH, can be classified into two types according to the EH model. For the first type, the terminal harvests radio frequency (RF) energy from the access point (AP) which usually integrates both the MEC server and energy transmitter [3]–[7]. In [3], a multiuser wireless powered MEC network was considered, where the AP broadcasted RF energy to distributed terminals. By jointly optimizing the individual mode selection and transmission time allocation, the sum computation rate was maximized. In [5], the authors studied the problem of maximizing the sum computation bits of all users in a backscatter-assisted MEC network.

While the second type of MEC-EH system adopts a more general EH model in which the terminal harvests ambient energies, including the solar energy, vibration of mechanical energy, RF energy, etc. [8]–[12]. Compared with the RF-type EH, the ambient type is much more random, leading to that it is difficult to be predicted. The system optimization objective turns to the time average of energy consumption or execution latency. The corresponding resources scheduling policy is also parameterized by time and referred to as *dynamic* scheduling. In [8], the authors proposed a Lyapunov optimization-based dynamic computation offloading algorithm, to minimize the time average latency, meanwhile stabilizing the battery energy queue. In [9], the authors studied the tradeoff between the energy consumption and execution delay for MEC systems with an energy queue and several task-buffer queues.

Currently, most works on MEC networks with EH focused on the single-antenna case except that only a few works studied the multi-input single-output (MISO) case, e.g. [6]. To the best of our knowledge, there has been no work on MIMO MEC with EH. Clearly, multiple antennas can provide the spatial diversity gain and the incorporation of MIMO is able to improve the transmission performance. On the other front, the ambient energy is easily accessible and does not require an energy transmitter. Motivated by the above, we study MIMO MEC networks with the ambient-type EH and the associated offloading strategy. The contributions and novelties are summarized as follows. 1) Model: Different from the models in existing works, we incorporate MIMO into MEC networks with EH; besides, partial offloading is adopted since it is more suitable for tasks with strict latency requirement. 2) Algorithm: The system design involves both the energy consumption and execution delay and aims at minimizing the long-time average of a weighted sum of them; a dynamic computation offloading (DCO) algorithm is proposed based on successive convex approximation (SCA) [13], [14] under the Lyapunov optimization framework. 3) Results: Multiple antennas and the optimization of transmission covariance matrix help reduce the system cost and the task drop ratio.

Notations: $\mathbb{E}(\cdot)$ denotes the statistical expectation; $\mathbb{C}^{m\times n}$ represents the space consisting of $m \times n$ complex matrices. For a matrix $\mathbf{X}$, the notations $\mathbf{X}^*$, $\mathbf{X}^H$, and $\mathrm{Tr}(\mathbf{X})$ denote its conjugate, Hermitian transpose, and trace, respectively; $\mathbf{I}_m$ is an $m \times m$ identity matrix.

II. System Model and Problem Formulation

The system consists of an MEC server and a mobile terminal which can harvest energy from surroundings. The MEC server and terminal are equipped with N_T and N_R antennas, respectively. Based on the channel condition, the terminal can offload the task partially onto the MEC server. Assume that the time is slotted and the length of each slot is τ. Denote the time slot set by $\Gamma \triangleq \{0, 1, \cdots\}$. At the beginning of each time slot, the task of length L in unit of bit arrives at the terminal with probability ρ, which is modelled as an *i.i.d* Bernoulli process. Further, assume that the task is response-time sensitive and should be finished within one time slot. Each task can be computed or simply dropped

Manuscript received October 19, 2020; revised February 11, 2021; accepted April 13, 2021. Date of publication April 27, 2021; date of current version June 9, 2021. This work was supported in part by the National Natural Science Foundation of China under Grants 61871139 and 61601275, in part by the International Science and Technology Cooperation Projects of Guangdong Province under Grant 2020A0505100060, and in part by the Natural Science Foundation of Guangdong Province under Grant 2021A1515011392. The review of this article was coordinated by Prof. Guido Marchetto. *(Corresponding author: Ling Xing.)*

Wen Zhou is with the College of Information Science and Technology, Nanjing Forestry University, Nanjing 210018, China (e-mail: wenzhou@ustc.edu).

Ling Xing is with the School of Information Engineering, Henan University of Science and Technology, Luoyang 471000, China (e-mail: xingling_my@haust.edu.cn).

Junjuan Xia and Lisheng Fan are with the School of Computer Science, Guangzhou University, Guangzhou 510006, China (e-mail: xiajunjuan@gzhu.edu.cn; lsfan@gzhu.edu.cn).

Arumugam Nallanathan is with the School of Electronic Engineering and Computer Science, Queen Mary University of London, London E1 4NS, U.K. (e-mail: a.nallanathan@qmul.ac.uk).

Digital Object Identifier 10.1109/TVT.2021.3075018

due to insufficient system resources, with an indicator $I_{\rm exe}(n) = 1(0)$ representing that the task is executed (discarded). If computed, the task can be separated into two parts: one part is computed locally while the other is offloaded and computed by the server.

A. Computation Model

At the n-th slot, assume that $\alpha(n)L$ bits of the task are executed locally and $(1-\alpha(n))L$ bits are offloaded to the MEC server.

1) Local Computation: The processing time for local computation is expressed as

$$D_l(n) = \alpha(n)L\eta/f(n), \tag{1}$$

where η is the number of cycles of processing one bit, and $f(n)$ is the CPU frequency at time slot n. The energy consumption is expressed as

$$E_l(n) = \alpha(n)\beta L\eta f^2(n), \tag{2}$$

where β is the coefficient that depends on the chip architecture.

2) MEC Computation: The processing time for MEC computation is given by

$$D_{\rm MEC}(n) = (1-\alpha(n))L/r(\mathbf{Q}(n)), \tag{3}$$

where $\mathbf{Q}(n)$ is the transmitter covariance matrix with $\mathrm{tr}(\mathbf{Q}(n)) \le P_T$ and P_T is the maximum transmission power. In (3), the uplink transmission rate is

$$r(\mathbf{Q}(n)) = B_W \log_2 \det(\mathbf{I}_{N_R} + \mathbf{H}(n)\mathbf{Q}(n)\mathbf{H}^H(n)/\sigma^2),$$

where $\mathbf{H}(n) \in \mathbb{C}^{N_R \times N_T}$ is the uplink channel matrix; W_B is the bandwidth of the MEC sever and σ^2 is the noise power. Note that both the feedback delay and MEC execution delay are omitted since they are usually small. The energy consumption for the mobile terminal is given by [15]

$$E_{\rm MEC}(n) = (1-\alpha(n))L\mathrm{tr}(\mathbf{Q}(n))/r(\mathbf{Q}(n)). \tag{4}$$

From (3) and (4), we can obtain an insight on the system.

3) System Latency and Energy Consumption: With (1), (2), (3), and (4), define the total time delay and energy consumption of the system as

$$D_{\rm sys}(n) \triangleq \mathbf{1}(I_{\rm exe}(n)=1) \cdot \max\left[D_l(n), D_{\rm MEC}(n)\right], \tag{5}$$

$$E_{\rm sys}(n) \triangleq \mathbf{1}(I_{\rm exe}(n)=1) \cdot \left[E_l(n) + E_{\rm MEC}(n)\right], \tag{6}$$

where $\mathbf{1}(.)$ is the indicator function.

B. Energy Harvesting Model

Assume that the arrived energy $E_H(n)$ is uniformly distributed in the interval $[0, E_H^{\rm MAX}]$. Clearly, the practical harvested energy $e(n)$ satisfies $0 \le e(n) \le E_H(n)$. The battery energy, denoted as $B(n)$, is updated according to the following equation

$$B(n+1) = B(n) + e(n) - E_{\rm sys}(n). \tag{7}$$

Note that $B(n)$ refers to the energy level at the beginning of time slot n [8], whereas $e(n)$ refers to the harvested energy during the period of time of length τ.

For the MEC system with the RF-type EH, the collected energy at slot n, $e(n)$, can be predicted and hence it can be used for the current time slot. However, for the system with the ambient EH, $e(n)$ can not be predicted and it is known at the end of slot n. Hence, the harvested energy $e(n)$ cannot be used at the current time slot n and can only be used for the next time slot. Consequently, the system energy consumption should satisfy

$$0 \le E_{\rm sys}(n) \le B(n). \tag{8}$$

C. Problem Formulation

The energy consumption and latency are two important factors in measuring the system performance, which are used to optimize the offloading policy. Due to the lack of battery energy or the deep fading of the channel, some tasks have to be dropped. Considering this, we introduce the task dropping cost Φ into the system cost function, which stands for a kind of 'penalization'. A large Φ means that the system prefers to execute the task, even if the channel condition is not good or the energy consumption is high, while a small Φ means the opposite. At time slot n, the system execution cost is defined as

$$F_{\rm cost}(n) = [E_{\rm sys}(n) + w_D D_{\rm sys}(n)] + \Phi \cdot \mathbf{1}(I_{\rm exe}(n)=0), \tag{9}$$

where w_D is the weight on the execution delay. Usually, we prefer executing a task to dropping it, which leads to $\Phi \ge \max[E_{\rm sys}(n) + w_D D_{\rm sys}(n)]$. Since $E_{\rm sys}(n) + w_D D_{\rm sys}(n) \le E_{\rm MAX} + w_D\tau$ holds, we may set $\Phi \ge E_{\rm MAX} + w_D\tau$. Then, the problem is formulated as P_1.

$$\begin{aligned} P_1: \min_{\{I_{\rm exe}(n),\mathbf{Q}(n),f(n),\alpha(n),e(n)\}} & \lim_{T\to\infty} \frac{1}{T}\mathbb{E}\left[\sum_{n=0}^{T-1} F_{\rm cost}(n)\right] \\ s.t.\ & C_1: D_{\rm sys}(n) \le \tau \\ & C_2: E_{\rm sys}(n) \le E_{\rm MAX} \\ & C_3: 0 \le E_{\rm sys}(n) \le B(n) \\ & C_4: 0 \le e(n) \le E_H(n) \\ & C_5: 0 \le \mathrm{tr}(\mathbf{Q}(n)) \le P_T \cdot \mathbf{1}(I_{\rm exe}(n)=1) \\ & C_6: 0 \le f(n) \le f_{\rm MAX} \cdot \mathbf{1}(I_{\rm exe}(n)=1) \\ & C_7: 0 \le \alpha(n) \le 1 \\ & C_8: I_{\rm exe}(n) \in \{0,1\}, n \in \Gamma. \end{aligned}$$

In P_1, C_1 is the latency constraint, C_2 and C_3 are energy constraints with $E_{\rm MAX}$ being the maximum energy consumption, C_5 is the transmission power constraint, and C_6 is the frequency constraint for local computation with $f_{\rm MAX}$ representing the maximum CPU frequency.

III. The Proposed Dynamic Computation Offloading Method

In this section, we first perform transformations on P_1 to make it satisfy the prerequisite of Lyapunov optimization. Then, we solve the transformed problem.

A. The DCO Algorithm

Lyapunov optimization technique is suitable to solve the problem involving time averages and queues. P_1 is just an example of this type. However, the technique cannot be applied to it directly. Because in C_3, $B(n)'s$ at different time slots are not independent so that the decision sets of P_1 at different time slots are related. This violates the prerequisite

Algorithm 1 : The Proposed DCO algorithm.

1: Initialize the total iteration number $T_{\max}$ and $n = 0$.
2: When the task arrives, obtain $B(n)$, $E_H(n)$, $\mathbf{H}(n)$, and compute $\tilde{B}(n)$.
3: Solve the following per-time slot problem.

$$P_3: \min_{\{I_{\text{exe}}(n),\mathbf{Q}(n),f(n),\alpha(n),e(n)\}} \tilde{B}(n)[e(n)-E_{\text{sys}}(n)]+VF_{\text{cost}}(n)$$
$$s.t.\ C_1, C_2', C_4 - C_8$$

4: Increase $n := n + 1$ and update $B(n)$.
5: If $n > T_{\max}$, end this procedure; otherwise repeat steps 2-5.

of Lyapunov optimization. To remove C_3, we introduce an energy lower bound E_{MIN} and construct a virtual energy queue $\tilde{B}(n)$ [8], [9].

First, introduce E_{MIN} into C_2, obtaining a modified version of P_1.

$$P_2: \min_{\{I_{\text{exe}}(n),\mathbf{Q}(n),f(n),\alpha(n),e(n)\}} \lim_{T\to\infty} \frac{1}{T}\mathbb{E}\left[\sum_{n=0}^{T-1} F_{\text{cost}}(n)\right]$$
$$s.t.\ C_1, C_3 - C_8$$
$$C_2': E_{\text{sys}}(n) \in \{0\} \cup [E_{\text{MIN}}, E_{\text{MAX}}]$$

For $E_{\text{MIN}} \to 0$, the optimal value of P_2 approaches that of P_1 [8], [9].

Second, construct a virtual energy queue $\tilde{B}(n) \triangleq B(n) - \theta$. Here, θ is a perturbation parameter satisfying

$$\theta \geq \tilde{E}_{\text{MAX}} + V(\Phi \cdot E_{\text{MIN}}^{-1} - 1), \tag{10}$$

where $\tilde{E}_{\text{MAX}} = \min(\beta L\eta f_{\text{MAX}}^2 + P_T\tau, E_{\text{MAX}})$ and V is a control parameter. By doing so, we can ignore C_3 and the resultant solution can automatically satisfy C_3, as will be proved later.

Third, omitting C_3 of P_2, we adopt the classic Lyapunov optimization to solve it. Note that $\tilde{B}(n)$ also satisfies (7) and the queue to be stabilized in P_2 is $\tilde{B}(n)$ instead of $B(n)$. Define the Lyapunov drift function as

$$\Delta(\tilde{B}(n)) = \frac{1}{2}\mathbb{E}[(\tilde{B}(n+1))^2 - (\tilde{B}(n))^2|\tilde{B}(n)].$$

Using similar technique in [8], [9], it is easy to derive that $\Delta(\tilde{B}(n)) \leq C_0 + \mathbb{E}[\tilde{B}(n)(e(n)-E_{\text{sys}}(n))|\tilde{B}(n)]$, where $C_0 = \frac{1}{2}[(E_H^{\text{MAX}})^2 + \tilde{E}_{\text{MAX}}^2]$. Consequently, the Lyapunov drift-plus-penalty function is upper bounded by

$$\Delta(\tilde{B}(n)) + V\left[F_{\text{cost}}(n)|\tilde{B}(n)\right] \leq C_0 + \Big\{\tilde{B}(n) \times [e(n) - E_{\text{sys}}(n)]\,|\tilde{B}(n)\Big\} + V\left[F_{\text{cost}}(n)|\tilde{B}(n)\right].$$

The Lyapunov optimization-based technique minimizes the upper bound of the drift-plus-penalty function per time slot so as to reduce the system cost, meanwhile keeping the queue $\tilde{B}(n)$ stable. Based on the above, the dynamic computation offloading algorithm is proposed and summarized in *Algorithm 1*. Further, we have the following two lemmas about this algorithm.

Lemma 1: The optimal solution of P_3 satisfies C_3.

Proof: To begin with, note that the optimization of $e(n)$ only involves C_4 and is not related with other variables. The optimal $e(n)$ can be obtained by solving the sub-problem of P_3:

$$P_{3-1}: \min \tilde{B}(n)e(n),\ s.t.\ C_4.$$

The optimal solution is $\bar{e}(n) = E_H(n) \cdot \mathbf{1}(\tilde{B}(n) \leq 0)$. Then, we consider two cases. For $B(n) \geq \tilde{E}_{\text{MAX}}$, it is clear that $E_{\text{sys}}(n) \leq \tilde{E}_{\text{MAX}}$ and hence $E_{\text{sys}}(n) \leq B(n)$. For $B(n) < \tilde{E}_{\text{MAX}}$, it is clear that $\tilde{B}(n) < 0$ and the objective of P_3 for executing the task is

$$\begin{aligned}
&\tilde{B}(n)\left[E_H(n) - E_{\text{sys}}(n)\right] + VE_{\text{sys}}(n) + Vw_D D_{\text{sys}}(n)\\
&= \tilde{B}(n)E_H(n) + \left[V - \tilde{B}(n)\right]E_{\text{sys}}(n) + Vw_D D_{\text{sys}}(n)\\
&> \tilde{B}(n)E_H(n) + V\Phi E_{MIN}^{-1}E_{MIN}\\
&= \tilde{B}(n)E_H(n) + V\Phi.
\end{aligned}$$

Note that $\tilde{B}(n)E_H(n) + V\Phi$ is the objective of P_3 for dropping the task. This means that we should drop this task, which results in $E_{\text{sys}}(n) = 0 \leq B(n)$. Therefore, *Lemma* 1 holds.

Lemma 2: $\bar{F}_{\text{cost}}^{P_2} \leq \bar{F}_{\text{cost}}^{\text{DCO}} \leq \bar{F}_{\text{cost}}^{P_2} + C_0/V$, where $\bar{F}_{\text{cost}}^{\text{DCO}}$ and $\bar{F}_{\text{cost}}^{P_2}$ are the average costs achieved by the DCO algorithm and solving problem P_2, respectively.

The proof of *Lemma* 2 can be referred to [8] and [16, Section 4.5]. Here, we will not go into more details.

From the proof of *Lemma 1* and *Lemma 2*, we can obtain the following insights on the system.

Remark 1: For $B(n) < \tilde{E}_{\text{MAX}}$, the task will be dropped. This means that the system keeps on discarding tasks in order to charge the battery initially. Besides, when $\tilde{B}(n) > 0$, i.e., $B(n) > \theta$, the system will discard the harvested energy. It is inferred that the battery energy is stable around the expected level θ, which will be verified by simulation results later.

Remark 2: Lemma 1 implies that the solution of P_3 is also feasible for P_1 and P_2. With *Lemma* 2, $\bar{F}_{\text{cost}}^{\text{DCO}} \to \bar{F}_{\text{cost}}^{P_2}$ when $V \to +\infty$. Recall that $\bar{F}_{\text{cost}}^{P_2} \to \bar{F}_{\text{cost}}^{P_1}$ when $E_{\text{MIN}} \to 0$. It follows that for $E_{\text{MIN}} \to 0$ and $V \to +\infty$, $\bar{F}_{\text{cost}}^{\text{DCO}} \to \bar{F}_{\text{cost}}^{P_1}$, where $\bar{F}_{\text{cost}}^{P_1}$ is the average cost of P_1. In other words, the proposed DCO method asymptotically achieves the optimal value of P_1.

B. The Algorithm of Solving the Per-Time Slot Problem P_3

In the DCO algorithm, the key step is to solve the per-time slot problem P_3. This subsection will develop a SCA based algorithm to solve this problem. From the previous subsection, the optimal $e(n)$ is obtained by solving P_{3-1} and the left is to optimize the task assignment. Since the solution for dropping the task is simple, we just consider the case of execution, which is expressed by another subproblem of P_3.

$$P_{3-2}: \min_{\{\mathbf{Q}(n),f(n),\alpha(n)\}} \left[-\tilde{B}(n) + V\right]E_{\text{sys}}(n) + Vw_D D_{\text{sys}}(n)$$
$$s.t.\ C_1, C_5 - C_7$$
$$C_2'': E_{\text{MIN}} \leq E_{\text{sys}}(n) \leq E_{\text{MAX}}$$

P_{3-2} is non-convex due to the objective function and energy constraint. Tackling it needs the introduction of auxiliary variables and some convex transformations. This problem can be divided into two cases according to whether $-\tilde{B}(n) + V \geq 0$.

1) $-\tilde{B}(N) + V \geq 0$*:* Introducing auxiliary variables S_1, S_2, and S_3, P_{3-2} can be equivalently written as

$$P_4: \min_{\{\mathbf{Q}(n),f(n),\alpha(n),S_1,S_2,S_3\}} \left[-\tilde{B}(n) + V\right](S_1 + S_2) + Vw_D S_3$$
$$s.t.\ C_{1-1}: \alpha(n)L\eta - f(n)S_3 \leq 0$$
$$C_{1-2}: (1-\alpha(n))\,L - r\left(\mathbf{Q}(n)\right)S_3 \leq 0$$
$$C_{1-3}: S_3 \leq \tau$$
$$C_{2-1}: \beta L\eta f^2(n)/S_1 \leq 1/\alpha(n)$$
$$C_{2-2}: (1-\alpha(n))\,\text{tr}\left(\mathbf{Q}(n)\right)/S_2 - r\left(\mathbf{Q}(n)\right)/L \leq 0$$

$$C_{2-3}: E_{\mathrm{MIN}} \le S_1 + S_2 \le E_{\mathrm{MAX}}$$
$$C_{2-4}: S_1 \ge 0, S_2 \ge 0$$
$$C_5': 0 \le \mathrm{tr}\left(\mathbf{Q}(n)\right) \le P_T$$
$$C_6': 0 \le f(n) \le f_{\mathrm{MAX}}$$
$$C_7: 0 \le \alpha(n) \le 1, n \in \Gamma$$

Observe that, $C_{1-1}, C_{1-2}, C_{2-1}$ and C_{2-2} are not convex and hence need convexification. Clearly,

$$f_1\left(\alpha(n), f(n)\right) \triangleq \frac{\alpha(n)}{f(n)}$$
$$\le \frac{1}{2}\left[b_1{}^{(k)}(\alpha(n))^2 + \frac{1}{b_1{}^{(k)}(f(n))^2}\right] \triangleq g_1\left(\alpha(n), f(n); b_1{}^{(k)}\right),$$

where $b_1^{(k)}$ denotes the value of b_1 in the k-th iteration. Using the convex upper bound of $f_1(\cdot)$, we replace C_{1-1} by the convex constraint C_{1-1}':

$$L\eta g_1\left(\alpha(n), f(n); b_1{}^{(k)}\right) \le S_3.$$

Similarly, we have the following inequalities.

$$f_2\left(\alpha(n), S_3\right) \triangleq \frac{1-\alpha(n)}{S_3}$$
$$\le \frac{1}{2}\left[b_2{}^{(k)}(1-\alpha(n))^2 + \frac{1}{b_2{}^{(k)}S_3^2}\right] \triangleq g_2\left(\alpha(n), S_3; b_2{}^{(k)}\right),$$
$$f_3\left(\alpha(n)\right) \triangleq \frac{1}{\alpha(n)}$$
$$\ge \frac{1}{b_3^{(k)}} - \frac{1}{\left[b_3^{(k)}\right]^2}\left(\alpha(n) - b_3^{(k)}\right) \triangleq g_3\left(\alpha(n); b_3^{(k)}\right),$$

and

$$f_4\left(\alpha(n), \mathbf{Q}(n), S_2\right) \triangleq (1-\alpha(n))\,\mathrm{tr}\left(\mathbf{Q}(n)\right)/S_2$$
$$\le \frac{1}{3}\left\{b_4^{(k)}(1-\alpha(n))^3 + b_5^{(k)}[\mathrm{tr}\left(\mathbf{Q}(n)\right)]^3 + \frac{1}{b_4^{(k)}b_5^{(k)}S_2^3}\right\}$$
$$\triangleq g_4\left(\alpha(n), \mathbf{Q}(n), S_2; b_4^{(k)}, b_5^{(k)}\right),$$

where $b_2^{(k)} \sim b_5^{(k)}$ have similar definitions to $b_1^{(k)}$. Using the inequalities above, we replace C_{1-2}, C_{2-1}, and C_{2-2} by the following convex constraints C_{1-2}', C_{2-1}', and C_{2-2}', respectively.

$$C_{1-2}': g_2(\alpha(n), S_3; b_2{}^{(k)}) \le \frac{r(\mathbf{Q}(n))}{L};$$
$$C_{2-1}': \beta L\eta f^2(n)/S_1 \le g_3(\alpha(n); b_3^{(k)});$$
$$C_{2-2}': g_4(\alpha(n), \mathbf{Q}(n), S_2; b_4^{(k)}, b_5^{(k)}) \le \frac{r(\mathbf{Q}(n))}{L}.$$

With $C_{1-1}', C_{1-2}', C_{2-1}'$ and C_{2-2}', we construct the convex problem $P_4^{(k)}$ that is parameterized by $b_1^{(k)} \sim b_5^{(k)}$.

$$P_4^{(k)}: \min_{\{\mathbf{Q}(n), f(n), \alpha(n), S_1, S_2, S_3\}} \left[-\tilde{B}(n) + V\right](S_1 + S_2) + V w_D S_3$$
$$s.t.\ C_{1-1}', C_{1-2}', C_{1-3}, C_{2-1}', C_{2-2}', C_{2-3}, C_{2-4}, C_5', C_6', C_7$$

Algorithm 2 : Solve Problem P_4.

1: Initialize $b_1^{(0)} \sim b_5^{(0)}$ and iteration index $k = 0$.
2: Solve $P_4^{(k)}$ and obtain the optimal solution: $\bar{\mathbf{Q}}(n)$, $\bar{f}(n)$, and $\bar{\alpha}(n)$.
3: $k := k + 1$.
4: Update $b_1^{(0)} \sim b_5^{(0)}$ according to (11). Note that we replace $\mathbf{Q}(n)$, $f(n)$, and $\alpha(n)$ with $\bar{\mathbf{Q}}(n)$, $\bar{f}(n)$, and $\bar{\alpha}(n)$ in (11), respectively.
5: Repeat steps 2-5 until convergence.

Note that the new constraints C_{1-1}', C_{1-2}', C_{2-1}' and C_{2-2}' are much tighter, therefore the feasible points of $P_4^{(k)}$ are also feasible for P_4.

Besides, when

$$b_1^{(k)} = 1/\left[\alpha(n) f(n)\right], \tag{11a}$$
$$b_2^{(k)} = 1/\left[(1-\alpha(n))\, S_3\right], \tag{11b}$$
$$b_3^{(k)} = \alpha(n), \tag{11c}$$
$$b_4^{(k)} = \mathrm{tr}\left(\mathbf{Q}(n)\right)/S_2/[1-\alpha(n)]^2, \tag{11d}$$
$$b_5^{(k)} = [1-\alpha(n)]/S_2/\left[\mathrm{tr}\left(\mathbf{Q}(n)\right)\right], \tag{11e}$$

the functions $f_1(\cdot) \sim f_4(\cdot)$ and $g_1(\cdot) \sim g_4(\cdot)$ satisfy [14, *Property A*], the prerequisite of the SCA-based method.

Based on the above, the SCA-based algorithm of solving P_4 is proposed and summarized in *Algorithm 2*. This algorithm solves a sequence of problems $\{P_4^{(k)}\}$. It is easy to verify that the optimal solution of $P_4^{(k)}$ is feasible for $P_4^{(k+1)}$. Thus, the objective obtained in the $(k+1)$-th iteration is less than or equal to that in the n-th iteration. Since the objective, no less than zero and hence lower bounded, decreases with n, this algorithm is convergent. Usually, the SCA-based algorithm converges to a good local optimal solution.

Remark 3: $b_3^{(k)}$ in C_{2-1}' is predetermined at the k-th iteration and should be treated as a constant; the function $f^2(n)/S_1$ is a quadratic-over-linear function, which is convex. Hence, C_{2-1}' is convex.

Remark 4: In P_4, the values of $S_1 \sim S_3$ and the objective function depend on a sequence of $P_4^{(k)}$ and they can be obtained after the convergence of this sequence. From a $P_4^{(k)}$ in a single iteration, the associated convergence values cannot be obtained.

2) $-\tilde{B}(N) + V < 0$: The processing steps in this part are similar to the previous ones. Introducing auxiliary variables S_1, S_2, and S_3, we transform P_{3-2} into an equivalent problem.

$$P_5: \min_{\{\mathbf{Q}(n), f(n), \alpha(n), S_1, S_2, S_3\}} \left[-\tilde{B}(n) + V\right](S_1 + S_2) + V w_D S_3$$
$$s.t.\ C_{1-1}, C_{1-2}, C_{1-3}, C_{2-3}, C_{2-4}, C_5', C_6', C_7$$
$$\bar{C}_{2-1}: \beta L\eta f^2(n)/S_1 \ge 1/\alpha(n)$$
$$\bar{C}_{2-2}: (1-\alpha(n))\,\mathrm{tr}\left(\mathbf{Q}(n)\right)/S_2 - r\left(\mathbf{Q}(n)\right)/L \ge 0$$

Clearly, the following inequalities hold.

$$f_5\left(S_1, \alpha(n)\right) \triangleq \sqrt{\frac{S_1}{\alpha(n)}} \le \frac{1}{2}\left[b_6^{(k)} S_1 + \frac{1}{b_6^{(k)}\alpha(n)}\right]$$
$$\triangleq g_5\left(S_1, \alpha(n); b_6^{(k)}\right),$$

$$f_6\left(S_2,\alpha\left(n\right),\mathbf{Q}\left(n\right)\right)\triangleq\frac{S_2}{\left(1-\alpha\left(n\right)\right)\mathrm{tr}\left(\mathbf{Q}\left(n\right)\right)}\leq\frac{1}{3}\left\{b_7^{(k)}S_2^3+\frac{b_8^{(k)}}{\left[1-\alpha\left(n\right)\right]^3}+\frac{\left[b_7^{(k)}b_8^{(k)}\right]^{-1}}{\left[\mathrm{tr}\left(\mathbf{Q}\left(n\right)\right)\right]^3}\right\}\triangleq g_6\left(S_2,\alpha\left(n\right),\mathbf{Q}\left(n\right);b_7^{(k)},b_8^{(k)}\right),$$

and

$$f_7\left(\mathbf{Q}\left(n\right)\right)\triangleq\frac{1}{r\left(\mathbf{Q}\left(n\right)\right)}\geq r\left(\mathbf{Q}_0^{(k)}\right)-\frac{1}{\left[r\left(\mathbf{Q}_0^{(k)}\right)\right]^2}\times\mathrm{tr}\left[\nabla_{\mathbf{Q}^*(n)}r\left(\mathbf{Q}_0^{(k)}\right)\left(\mathbf{Q}\left(n\right)-\mathbf{Q}_0^{(k)}\right)\right]\triangleq g_7\left(\mathbf{Q}\left(n\right);\mathbf{Q}_0^{(k)}\right),$$

where $\nabla_{\mathbf{Q}^*(n)}r(\cdot)$ is the gradient of $r(\cdot)$ with respect to the conjugate of $\mathbf{Q}(n)$ and

$$\nabla_{\mathbf{Q}^*(n)}r\left(\mathbf{Q}_0^{(k)}\right)=\frac{B_W\mathbf{H}^H(n)}{\ln 2\cdot\sigma^2}\left[\mathbf{I}+\frac{\mathbf{H}(n)\mathbf{Q}_0^{(k)}\mathbf{H}^H(n)}{\sigma^2}\right]^{-1}\mathbf{H}(n).$$

Note that, since both $\nabla_{\mathbf{Q}^*(n)}r(\cdot)$ and $(\mathbf{Q}(n)-\mathbf{Q}_0^{(k)})$ are Hermitian, it is easy to verify that $\mathrm{tr}[\nabla_{\mathbf{Q}^*(n)}r(\mathbf{Q}_0^{(k)})\times(\mathbf{Q}(n)-\mathbf{Q}_0^{(k)})]$ is a real number.

We use the above inequalities to convexify $\bar{C}_{2-1}$ and $\bar{C}_{2-2}$, obtaining

$$\bar{C}'_{2-1}:\sqrt{\beta L\eta}f(n)\leq g_5\left(S_1,\alpha(n);b_6^{(k)}\right)$$
$$\bar{C}'_{2-2}:g_6\left(S_2,\alpha(n),\mathbf{Q}(n);b_7^{(k)},b_8^{(k)}\right)\geq L\cdot g_7\left(\mathbf{Q}(n);\mathbf{Q}_0^{(k)}\right).$$

With C_{1-1},' C_{1-2},' $\bar{C}'_{2-1}$ and $\bar{C}'_{2-2}$, construct the convex problem $P_5^{(k)}$, parameterized by $b_1^{(k)}, b_2^{(k)}, b_6^{(k)}\sim b_8^{(k)}$, and $\mathbf{Q}_0^{(k)}$.

$$P_5^{(k)}:\min_{\{\mathbf{Q}(n),f(n),\alpha(n),S_1,S_2,S_3\}}\left[-\tilde{B}(n)+V\right](S_1+S_2)+Vw_DS_3$$
$$s.t.\ C'_{1-1},C'_{1-2},C_{1-3},\bar{C}'_{2-1},\bar{C}'_{2-2},C_{2-3},C_{2-4},C'_5,C'_6,C_7$$

Besides, when (11a)–(11b) hold and

$$b_6^{(k)}=1/\sqrt{S_1\alpha(n)},\tag{12a}$$
$$b_7^{(k)}=1/\left[\left(1-\alpha(n)\right)\mathrm{tr}\left(\mathbf{Q}(n)\right)S_2^2\right],\tag{12b}$$
$$b_8^{(k)}=S_2(1-\alpha(n))^2/\mathrm{tr}\left(\mathbf{Q}(n)\right),\tag{12c}$$
$$\mathbf{Q}_0^{(k)}=\mathbf{Q}(n),\tag{12d}$$

the functions $f_5(\cdot)\sim f_7(.)$ and $g_5(\cdot)\sim g_7(.)$ satisfy [14, *Property A*].

Based on the above, it is easy to design the algorithm to solve P_5. Since it is similar to *Algorithm 2* in structure, we omit the details. The main differences from *Algorithm 2* lie in two aspects: 1) Step 1 initializes $b_1^{(0)}, b_2^{(0)}, b_6^{(0)}\sim b_8^{(0)}, \mathbf{Q}_0^{(k)}$ instead of $b_1^{(0)}\sim b_5^{(0)}$; 2) Step 4 updates these variables according to (11a), (11b), and (12). Finally, we present the algorithm of solving the per-time slot problem P_3, summarized in *Algorithm 3*.

Remark 5: The problem P_4 can be solved by the block coordinate descent (BCD) approach, by which the coupling among variables of P_4 can be tackled. However, it can not solve P_5 — another subproblem of the per-time slot problem P_3, where the main obstacle lies in the constraint $\bar{C}_{2-2}$. Because when optimizing $\mathbf{Q}(n)$, the constraint $\bar{C}_{2-2}$ is not convex even if we fix all other variables. In summary, the BCD approach can solve P_4 but not P_5, leading to that it cannot solve P_3. To give a uniform framework for the solution of problems P_3, P_4 and P_5, we adopt the SCA-based algorithm in this work.

Algorithm 3 : SCA-based Algorithm for the Per-time Slot Problem P_3.

1: Obtain the optimal harvested energy $\bar{e}(n)=E_H(n)\cdot\mathbf{1}(\tilde{B}(n)\leq 0)$.
2: If $-\tilde{B}(n)+V\geq 0$, solve P_4 by *Algorithm 2*; otherwise, solve P_5 by the similar algorithm. Denote the optimal solution and its objective as $\{\bar{\mathbf{Q}}(n),\bar{f}(n),\bar{\alpha}(n)\}$ and $\bar{F}_{\mathrm{obj}}$, respectively.
3: If $\bar{F}_{\mathrm{obj}}<V\Phi$, set the optimal $\bar{I}_{\mathrm{exe}}(n)=1$ and output $\{\bar{I}_{\mathrm{exe}}(n),\bar{\mathbf{Q}}(n),\bar{f}(n),\bar{\alpha}(n),\bar{e}(n)\}$; otherwise $\bar{I}_{\mathrm{exe}}(n)=0$ and output $\{\bar{I}_{\mathrm{exe}}(n),\bar{e}(n)\}$.

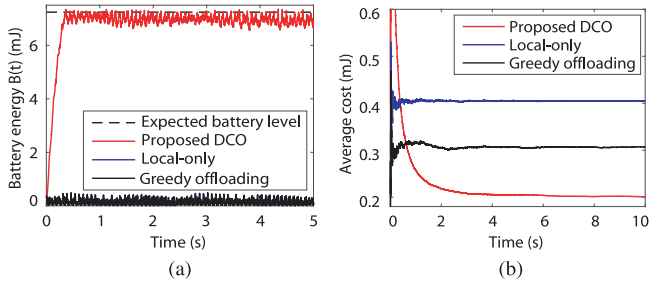

Fig. 1. Battery energies and average costs of several schemes v.s. time.

IV. Simulation Results

In this section, computer simulation is deployed to investigate the performance of the proposed algorithm. The uplink channel is modeled as $\mathbf{H}(n)=\frac{0.01}{d^2}\mathbf{H}_\omega(n)$, where d is the distance between transmitter and receiver and set to 80 m; $\mathbf{H}_\omega(n)$ is a random matrix with *i.i.d.* zero-mean and unit variance complex Gaussian random variables. For the mobile terminal, the task of length $L=2000$ bits arrives with probability $\rho=0.4$; the arrived energy is uniformly distributed in $0\sim E_H^{\mathrm{MAX}}=120\ \mu$J. Besides, $\beta=10^{-28}$ J $\cdot$ s^2, $\tau=2$ ms, $E_{\mathrm{MAX}}=1$ mJ, $\Phi=2$ mJ, $B_W=1$ MHz, $P_T=1$ W, $f_{\mathrm{MAX}}=1.5$ GHz, $\eta=750$ cycles/bit, the noise power $\sigma^2=3.4\times10^{-13}$ W. Unless otherwise specified, $N_T=N_R=2$, the weight factors $V=0.16$ mJ and $w_D=0.5$ W, $E_{\mathrm{MIN}}=0.05$ mJ, and the expected battery level $\theta=\tilde{E}_{\mathrm{MAX}}+V(\Phi\cdot E_{\mathrm{MIN}}^{-1}-1)$.

Five benchmark schemes are included for comparison: the local-only scheme [9], MEC-only scheme [9], greedy offloading scheme [9], isotropic transmission, and the 'DCO-f' scheme. The isotropic transmission means the proposed DCO with fixed $\mathbf{Q}(n)=\sqrt{P_T}\mathbf{I}$; the 'DCO-f' scheme refers to the proposed DCO with fixed $f(n)=f_{\mathrm{MAX}}$.

Observe from fig. 1(a) that the battery energy of the proposed DCO increases in the first 0.4 s and keeps stable around the expected battery level. For the other two schemes, their energies fluctuate at a lower level due to lack of effective energy management. In fig. 1(b), the average cost of DCO converges to 0.2 mJ, lower than the costs achieved by the local-only and greedy offloading schemes. However, its convergence speed is slow because the battery energy management forces many tasks to be abandoned at the beginning, resulting in a high initial cost. In Fig. 2, with increasing V and decreasing E_{MIN}, DCO's cost decreases and it converges to around 0.2 mJ eventually, which verifies DCO's asymptotic optimality mentioned in *Lemma 2*. Given appropriate V and E_{MIN}, e.g., $V\geq0.1$ and $E_{\mathrm{MIN}}\leq0.1$, the proposed DCO is able to achieve the lowest average cost among the six schemes.

Table I presents the task drop ratios under six schemes. When the number of antennas increases, resulting in more channel gains, the drop ratios for the proposed DCO, istropic, DCO-f, MEC-only and Greedy

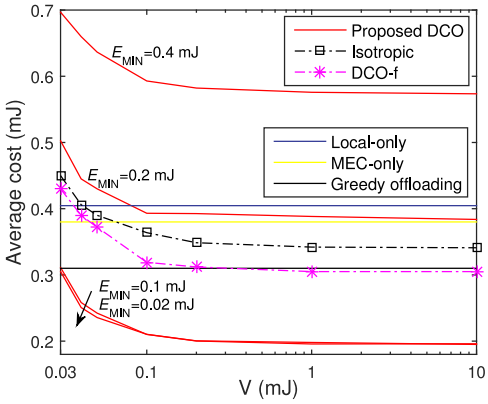

Fig. 2. Average cost v.s. V.

TABLE I
TASK DROP RATIOS OF DIFFERENT SCHEMES

Schemes	Number of antennas ($N_T = N_R$)		
	1	2	3
DCO	0.0020	0	0
Local-only	0.090	0.090	0.090
MEC-only	0.66	0.26	0.043
Greedy	0.088	0.061	0.0075
Isotropic	0.0040	0	0
DCO-f	0.26	0.15	0

offloading decrease. Clearly, the drop ratio of the DCO is always the lowest among the six schemes.

To sum up, the proposed DCO is superior to other schemes in terms of the average cost and drop ratio performance.

V. CONCLUSION

In this letter, we proposed a DCO scheme for MIMO MEC-EH systems, where the offloading ratio, transmission covariance matrix, and CPU-cycle frequency were jointly optimized to minimize the average weighted sum of energy consumption and latency. Further, two lemmas were presented to support the proposed scheme. Simulation results show that, given appropriate parameters, the proposed DCO algorithm has better performance than the benchmark schemes, in terms of both the system cost and task drop ratio. For the future work, we would like to extend the findings in this work to the scenarios with multiple edge users or with complicated noise [17].

REFERENCES

[1] C. Li, "Dynamic offloading for multiuser muti-CAP MEC networks: A deep reinforcement learning approach," *IEEE Trans. Veh. Technol.*, vol. 70, no. 3, pp. 2922–2927, Mar. 2021.
[2] G. Gui, M. Liu, F. Tang, N. Kato, and F. Adachi, "6G: Opening new horizons for integration of comfort, security, and intelligence," *IEEE Wireless Commun.*, vol. 27, no. 5, pp. 126-132, Oct. 2020.
[3] S. Bi and Y. J. Zhang, "Computation rate maximization for wireless powered mobile-edge computing with binary computation offloading," *IEEE Trans. Wireless Commun.*, vol. 17, no. 6, pp. 4177–4190, Jun. 2018.
[4] Q. Gu, Y. Jian, G. Wang, R. Fan, H. Jiang, and Z. Zhong, "Mobile edge computing via wireless power transfer over multiple fading blocks: An optimal stopping approach," *IEEE Trans. Veh. Technol.*, vol. 69, no. 9, pp. 10348–10361 , Sep. 2020.
[5] L. Shi, Y. Ye, X. Chu, and G. Lu, "Computation bits maximization in a backscatter assisted wirelessly powered MEC network," *IEEE Commun. Lett.*, vol. 25, no. 2, pp. 528-532, Feb. 2021.
[6] F. Wang, J. Xu, X. Wang, and S. Cui, "Joint offloading and computing optimization in wireless powered mobile-edge computing systems," *IEEE Trans. Wireless Commun.*, vol. 17, no. 3, pp. 1784–1797, Mar. 2018.
[7] F. Wang, J. Xu, and S. Cui, "Optimal energy allocation and task offloading policy for wireless powered mobile edge computing systems," *IEEE Trans. Wireless Commun.*, vol. 19, no. 4, pp. 2443–2459, Apr. 2020.
[8] 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.
[9] 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. Inform.*, vol. 14, no. 10, pp. 4642–4655, Oct. 2018.
[10] M. Merluzzi, P. D. Lorenzo, and S. Barbarossa, "Latency-constrained dynamic computation offloading with energy harvesting IoT devices," in *Proc. IEEE INFOCOM Conf. Comput. Commun.*, 2019, pp. 750–755.
[11] Y. Deng, Z. Chen, X. Yao, S. Hassan, and A. M. A. Ibrahim, "Parallel offloading in green and sustainable mobile edge computing for delay-constrained IoT system," *IEEE Trans. Veh. Technol.*, vol. 68, no. 12, pp. 12202–12214 , Dec. 2019.
[12] Z. Wei, B. Z., J. Su, and X. Lu, "Dynamic edge computation offloading for Internet of Things with energy harvesting: A learning method," *IEEE Internet Things J.*, vol. 6, no. 3, pp. 4436–4447, Jun. 2019.
[13] G. Scutari, F. Facchinei, and L. Lampariello, "Parallel and distributed methods for constrained nonconvex optimization—Part I: Theory," *IEEE Trans. Signal Process.*, vol. 65, no. 8, pp. 1929–1944, Apr. 2017.
[14] A. Beck, A. Ben-Tal, and L. Tetruashvili, "A sequential parametric convex approximation method with applications to nonconvex truss topology design problems," *J. Glob. Optim.*, vol. 47, no. 1, pp. 29–51, May 2010.
[15] S. Sardellitti, G. Scutari, and S. Barbarossa, "Joint optimization of radio and computational resources for multicell mobile-edge computing," *IEEE Trans. Signal Inf. Process. Netw.*, vol. 1, no. 2, pp. 89–103, Jun. 2015.
[16] M. J. Neely, *Stochastic Network Optimization With Application to Communication Queueing Systems*. San Rafael, CA, USA: Morgan Calypool, 2010.
[17] K. He, "Learning based signal detection for MIMO systems with unknown noise statistics," *IEEE Trans. Commun.*, vol. 69, no. 5, pp. 3025–3038, May 2021.