Energy-Efficient Cooperative Offloading for Multi-AP MEC in IoT Networks

Zhihui Cao[1], Haifeng Sun[1(✉)], Ning Zhang[2], and Xiang Lv[1]

[1] School of Computer Science and Technology, Southwest University of Science and Technology, Mianyang 621010, China
[2] University of Windsor, Windsor, Canada

Abstract. Mobile Edge Computing (MEC) technology is used for offloading local application tasks on Mobile Devices (MDs) to the edge server to decrease task processing time and reduce energy consumption in Internet of Things (IoTs) networks. In this paper, we investigate a scenario consisting of a local MD adjacent with a group of other MDs, one of which can act as the offloading cooperator. All the MDs are surrounded by multiple Access Points (APs), and each AP is deployed an MEC server providing abundant computation resources. Based on this scenario, we propose a cooperative energy-efficient offloading scheme under delay constraint. The local MD can offload part of the application task to a cooperative relay MD or the MEC server, and the relay MD can also offload part of the segment to an AP. By solving the proposed energy-efficient cooperative offloading problem under the constraint of computing delay, the most energy-efficient cooperative offloading MD and the AP as well as the task segmentation to minimize the energy consumption are determined. Numerical analysis shows that our proposed scheme significantly outperforms the benchmark schemes in the aspect of energy consumption and the supported task length in maximum.

Keywords: Mobile Edge Computing · Cooperative offloading · Multiple-AP · Energy efficient

1 Introduction

The continuous development of Internet of Things (IoTs) technology in the fifth generation (5G) communication systems has boosted an impact on a variety of applications in different fields. The IoT integrating people's lives and work, has a huge impact on the world economy, military, politics, culture and other aspects, and makes people's way of life change dramatically. The future world will be interconnected and globally intelligent [11]. Industrial IoT applications are in the explosive growth stage, on the other hand the computing power and latency feedback required by these applications are extremely strict and require a certain

This work was supported in part by the Applied Basic Research Programs of Science & Technology Committee Foundation of Sichuan Province (2019YJ0309).

H. Gao and X. Wang (Eds.): CollaborateCom 2021, LNICST 407, pp. 3–17, 2021.
https://doi.org/10.1007/978-3-030-92638-0_1

amount of massive application resources to handle computation-intensive application tasks. At present, the high-performance Central Processing Unit (CPU) chips on the terminal devices keep on constantly being updated, but still can't meet the requirement of dealing the huge application tasks under a limited delay constraint. The quite finite battery life of the Mobile Device (MD) is also a difficult problem to solve today. The energy consumption required by a large number of computing undoubtedly shortens the working time of the MD, which greatly affects the user experience [2]. Cloud computing can transfer the application tasks to the servers in the cloud center [16]. However, a large number of MDs in different area locations will cause heavy bandwidth load and high transmission delay, which makes cloud computing incompetent for the delay-sensitive applications.

Mobile Edge Computing (MEC) is a promising technology. In recent years, mobile computing has been shifting from centralized cloud computing to MEC, driven by 5G communication technologies. The main feature of MEC is offloading computation-intensive and high-latency constraint application data to the edge of the network near the MD, as well as reducing the pressure on the cloud-centric server, which significantly reduces energy consumption and computing latency at MDs [9].

In the MEC-based IoT network, we consider a scenario of an MD adjacent with many other MDs, one of which can act as the offloading cooperator. All the MDs are surrounded by a group of Access Points (APs), and each AP is deployed an MEC server providing abundant computation resources. It is of great significance to select a neighboring MD as the offloading relay to optimize the energy consumption in the whole task computing process with delay constraints [10]. In some cases, an application task is supported to be divided into several segments in different sizes, so we can select multiple adjacent relay MDs for offloading. But there are other cases where the application task is indivisible, or only supported to be divided into limited number of segments, so we can select a most energy-efficient relay MD for cooperative offloading. According to some research results that not all the task offloading can save energy [7]. Communication links, communication modes and so on can all affect the energy consumption. If an MD has a long distance from the AP accompanied by poor communication channel, offloading of tasks will use further energy than processing them locally. However, some studies have shown that the use of cooperative offloading by relay MDs can save communication energy [17].

We propose an efficient cooperative task offloading scheme for multi-AP MEC under delay constraint in the IoT network. By selecting an appropriate relay MD and the AP as well as the task segmentation for cooperative task offloading, the energy consumption of the whole task computing process is minimized. For this purpose, we first setup the system model, then we present the task offloading process with the relay MD. Next, we give the computation latency and energy consumption for each process on the local MD, the relay MD as well as the MEC server, respectively. Therefore we get a minimum energy consumption optimization problem in total for the whole process of task computing and offloading,

prove the problem is convex and solve it by the convex optimization toolbox CVX [4]. Finally, the conducted simulation results indicate the proposed scheme outperforms the benchmark schemes. The contribution of this paper is summarized below.

1) We consider a multi-AP offloading scenario that a local MD is adjacent with a group of other MDs, one of which can act as the offloading cooperator. All the MDs are surrounded by some APs, and each AP is deployed an MEC server with abundant computation resources. An application task can be partitioned into finite segments, and one adjacent MD can be selected as the collaborative offloading relay MD.
2) A scheme is proposed with energy-efficient cooperative task offloading for multi-AP MEC under delay constraint in IoT networks. The scheme minimizes the energy consumption under delay constraint by selecting the appropriate relay MD and the AP for cooperative task offloading.
3) By solving the optimization problem, the selection of the relay MD and the AP as well as the task segmentation were determined. A large number of numerical experimental results verified the advantages of our proposed scheme.

The rest of this paper is described as follows. In Sect. 2, we review MEC works in the MEC-based IoT. Section 3 presents the system model. Section 4 formulates the delay constrained energy minimization problem and presents the optimal solutions. Section 5 performs the numerical experiments. We conclude our work in Sect. 6.

2 Related Work

MEC can effectively reduce computing latency in computation-intensive applications. When processed by local MDs, delay sensitive tasks have to take a lot of time to wait, and the user demand is not satisfied [20]. Most of today's MDs are hardly to meet the computing requirements of a large number of delay-sensitive applications. The traditional method is to offload the application tasks to the cloud center server for computing. However, faced with numerous IoT devices, the cloud center server bears too much pressure, which greatly increases the task transmission time and computing latency. But in the MEC mode, the offloading of application tasks to the network edge for computing reduces the pressure at the cloud-centric server, solves the problem of high task transmission delay, and attracts widespread concern in academia and industry [19].

Some studies have shown that co-offloading can help reduce energy consumption compared with direct offloading. Baidas *et al.* maximized the network and offloading efficiency of all user clusters through power allocation and computing resource allocation [1]. Sun *et al.* introduced a scenario where an MD is surrounded by multiple MDs and an AP integrated with an MEC server, and proposed an optimal neighbor aided cooperative offloading scheme [15]. Li *et al.* proposed an artificial intelligence (AI) based single AP collaborative offloading

computing approach to determine the task offloading, computing, and result delivery policy [8]. Wei *et al.* studied single AP partial offloading of application tasks. The authors in [18] proposed an energy-saving optimization problem of MDs with separable tasks, and solved it using greedy algorithm. Pan *et al.* proposed an iterative algorithm implemented by successive convex approximation, and got the task offloading partition and time allocation [13]. But these works only support one AP scenario. Fan *et al.* investigated the cooperations of multiple mobile edge computing enabled-base station (MEC-BS), and proposed a novel scheme to enhance the computation offloading service of a MEC-BS through further offloading the extra tasks to other MEC-BSs connected to it [3], but this work did not consider cooperative offloading. Different from previous works, we investigate the scenario of an local MD adjacent with a group of other MDs, one of which can act as the offloading cooperator. Some APs randomly locate around, and each AP deployed an MEC server with abundant computation resources. The optimization problem with minimized energy consumption in the multi-AP offloading process, the selection of the offloading cooperator and APs, as well as the optimized task segmentation were solved.

3 System Model

In this section, firstly we set up an application scenario in the IoT network. Then, the delay constraint and energy consumption of an MD in local processing, relay MD processing and MEC server processing phases are respectively explored. Because of the particularity of application tasks, we consider tasks can be divided only into limited segments, and only one MD can be selected as a cooperative offloading assistant.

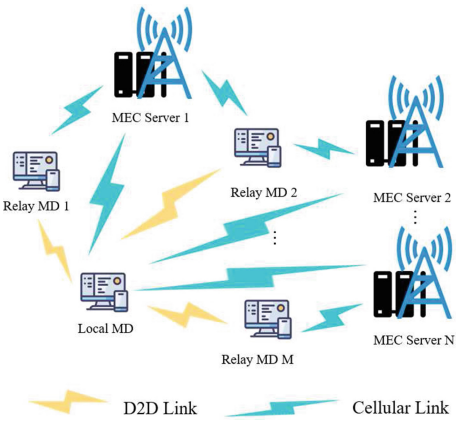

Fig. 1. System model.

As shown in Fig. 1, we set up a local MD u adjacent to multiple MDs $\mathcal{M} = \{1, 2, \ldots, M\}$, one of which can act as an relay MD $m_i, i \in \mathcal{M}$ to help the local MD u offload application tasks to the randomly surrounded APs $\mathcal{N} = \{1, 2, \ldots, N\}$, each one deployed an MEC server, which provides a abundant of computing resources. The offloading collaborative MD m_i can computing the offloaded segment locally, or select one AP $s_j, j \in \mathcal{N}$ deployed with an MEC server for offloading the sub-segment through cellular links. On the other hand, the task segment in the local MD u can also be directly offloaded to the AP $s_k, k \in \mathcal{N}$ through cellular links. In summary, each segment of the task can be processed on the local MD, the relay MD and the MEC server, separately [12]. The D2D link is used to transmit the offloading data between MDs. After the task computing is completed, the result will be returned to the local MD. Each MD can request the cooperative offloading, and we can consider the scenario of one task cooperative offloading.

In this network, we consider that cellular links as well as D2D links are deployed at different frequencies, so that all nodes do not affect each other when transferring data. We assume that the channel follows the decline of quasi-static blocks. In other words, during the offloading of a task segment, the channel state keeps constant [6]. In addition, due to the short length of the result, the delay and energy consumption of the result feedback are not considered.

We denote the application task length as $L > 0$, and the delay constraint as $T \geq 0$. For the total task length L of applications, it can be divided into four different segments. Let $l_u \geq 0, l_u^{s_k} \geq 0, l_u^{m_i} \geq 0$ and $l_{m_i}^{s_j} \geq 0$ represent the length of the computing segment at the local MD u, offloading from u to the AP s_k, offloading from u to the relay MD m_i, and offloading from m_i to the AP s_j, respectively. Then we have

$$L = l_u + l_u^{s_k} + l_u^{m_i} + l_{m_i}^{s_j}. \tag{1}$$

In particular, the task segment performed on the relay MD m_i is $l_u^{m_i}$, and the task segment $l_{m_i}^{s_j}$ is offloaded from m_i to the MEC server s_j, so the total task segment length that is offloaded from the local MD u to the relay MD m_i is $l_u^{m_i} + l_{m_i}^{s_j}$.

The whole process of task computing and offloading in the model include three phases. The first is the local processing phase, the second is the relay MD cooperative offloading processing phase, and the third is the MEC server processing phase. We will explore the computing delay, the offloading delay, together with energy consumption at each phase in detail.

3.1 Local Processing Phase

The local processing phase includes the self-computing of segment l_u at the local MD u, offloading one segment $l_u^{s_k}$ from u to the AP s_k, and offloading one segment $l_u^{m_i} + l_{m_i}^{s_j}$ from u to the relay MD m_i.

Local Computing. During the local processing phase, the segment of l_u will be executed on the local MD. Let c_u represent the number of local MD CPU cycles used for each bit, f_u represent the maximum computing capability (CPU cycles/second) of the local MD CPU. f'_u represent the computing capability required by the local MD for l_u. Then the computing time of l_u is

$$T_u^C = \frac{c_u l_u}{f'_u}. \tag{2}$$

Because the task should be executed under the delay constraint T, the task segment of each phase will be completed within T, namely the processing time T_u^C of l_u need to meet the delay constraint $T \geq T_u^C$. Put Eq. (2) into the constraint and we get $\frac{c_u l_u}{f'_u} \leq T$, and because $f'_u \leq f_u$, we get that the time constraint for the local processing phase is

$$\frac{c_u l_u}{f_u} \leq T. \tag{3}$$

The energy consumption E_u^C for the local MD computing can be described as

$$E_u^C = \gamma_u c_u f_u'^2 l_u, \tag{4}$$

where γ_u is the effective CPU capacitance coefficient on the local MD. For achieving the minimum energy consumption at the local MD, we set the computing time T_u^C equal to the delay constraint T, thus we obtain $T = T_u^C = \frac{c_u l_u}{f'_u}$, i.e. $f'_u = \frac{c_u l_u}{T}$. Substituting it into Eq. (4), we get

$$E_u^C = \frac{\gamma_u c_u^3 l_u^3}{T^2}. \tag{5}$$

Offloading to the MEC Server for Processing. The local MD offloads part of the task $l_u^{s_k}$ to the MEC server s_k through cellular links, in which the transmission power of the MD is expressed as $P_{u,s_k} \geq 0$. Suppose the channel power gain between them be $h_{u,s_k} \geq 0$, $\sigma_{s_k}^2$ denote the noise power at the AP intergrated with s_k and $\omega_k, k \in \mathcal{N}$ denote the cellular channel bandwidth from the local MD to the AP, so the data transmission rate (bits/second) from the local MD to the MEC server s_k is

$$r\left(P_{u,s_k}\right) = \omega_k \log_2\left(1 + \frac{p_{u,s_k} h_{u,s_k}}{\sigma_{s_k}^2}\right). \tag{6}$$

By Eq. (6), we can get the offloading delay T_{u,s_k}^O and offloading energy consumption E_{u,s_k}^O from the local MD to the MEC server s_k as

$$T_{u,s_k}^O = \frac{l_u^{s_k}}{r\left(P_{u,s_k}\right)}, \tag{7}$$

$$E_{u,s_k}^O = \frac{P_{u,s_k} l_u^{s_k}}{r\left(P_{u,s_k}\right)}. \tag{8}$$

Offloading to the Relay MD for Cooperative Processing. The relay MD m_i receives the segment with the length of $l_u^{m_i}+l_{m_i}^{s_j}$ sent by the local MD through D2D links on the transmit power $P_{u,m_i} \geq 0$. Let the channel power gain from the local MD to the relay MD m_i be $h_{u,m_i} \geq 0$, $\sigma_{m_i}^2$ denote the noise power at the relay MD m_i, and $\omega_i, i \in \mathcal{M}$ denote the cellular channel bandwidth from the local MD to the relay MD m_i, then the data transmission rate (bits/second) offloading from the local MD to the relay MD m_i is

$$r\left(P_{u,m_i}\right) = \omega_i \log_2\left(1 + \frac{p_{u,m_i} h_{u,m_i}}{\sigma_{m_i}^2}\right). \tag{9}$$

From Eq. (9), we can get the offloading delay T_{u,m_i}^O from the MD to the relay MD m_i and the energy consumption E_{u,m_i}^O for offloading as

$$T_{u,m_i}^O = \frac{l_u^{m_i} + l_{m_i}^{s_j}}{r\left(P_{u,m_i}\right)}, \tag{10}$$

$$E_{u,m_i}^O = \frac{P_{u,m_i}\left(l_u^{m_i} + l_{m_i}^{s_j}\right)}{r\left(P_{u,m_i}\right)}. \tag{11}$$

3.2 Relay MD Processing Phase

The relay MD cooperative offloading processing phase consists of two parts, one is the relay MD m_i computing of the segment $l_u^{m_i}$ that is offloaded from the local MD, and the other is the segment $l_{m_i}^{s_j}$ that is offloaded from m_i to the MEC server s_j.

Computing at the Relay MD. Choosing the properly relay MD with minimal energy consumption for offloading is significant [5,14,21]. We use c_{m_i} to represent the number of CPU cycles of each computing bit at the relay MD m_i, f_{m_i} to represent the maximum CPU computing capacity of the relay MD m_i, and $f_{m_i}^{'}$ to represent the computing capacity required by m_i according to the application segment. Thus, the computing delay at the relay MD m_i is

$$T_{u,m_i}^C = \frac{c_{m_i} l_u^{m_i}}{f'_{m_i}}. \tag{12}$$

For the relay MD m_i, the delay constraint includes the offloading delay from the local MD and the segment computing time, so we have

$$T_{u,m_i}^O + T_{u,m_i}^C \leq T. \tag{13}$$

Same as the local processing phase, since $f_{m_i} \geq f'_{m_i}$, substitute Eq. (10) and Eq. (12) into Eq. (13), we obtain the delay constraint of the relay MD m_i as

$$\frac{l_u^{m_i} + l_{m_i}^{s_j}}{r\left(P_{u,m_i}\right)} + \frac{c_{m_i} l_u^{m_i}}{f_{m_i}} \leq T. \tag{14}$$

The energy consumption for computing at the relay MD m_i is

$$E_{m_i}^C = \gamma_{m_i} c_{m_i} f_{m_i}'^2 l_{m_i}, \tag{15}$$

where γ_{m_i} is the effective capacitance coefficient at the CPU of the relay MD m_i. Substitute Eq. (12) into Eq. (15), similar to Eq. (5), we can obtain the energy consumption for computing at the relay MD m_i as

$$E_{m_i}^C = \frac{\gamma_{m_i} c_{m_i}^3 {l_u^{m_i}}^3}{\left(T - T_{u,m_i}^O\right)^2}. \tag{16}$$

Offloading to the MEC Server. The relay MD m_i offloads the segment $l_{m_i}^{s_j}$ through cellular links to the MEC server with sufficient computing resources for computing. Let the transmission power of the relay MD m_i offloading to the AP integrated with an MEC server s_j be $P_{m_i,s_j} \geq 0$, the channel power gain be $h_{m_i,s_j} \geq 0$, the noise power at the AP be $\sigma_{s_j}^2$, and $\omega_j, j \in \mathcal{N}$ denote the cellular channel bandwidth from the relay MD m_i to the AP integrated with an MEC server s_j. Therefore, the data transmission rate (bits/second) from the relay MD m_i to the MEC server s_j is

$$r\left(P_{m_i,s_j}\right) = \omega_j \log_2\left(1 + \frac{p_{m_i,s_j} h_{m_i,s_j}}{\sigma_{s_j}^2}\right). \tag{17}$$

Then, from Eq. (17), we can describe the offloading delay T_{m_i,s_j}^O of the relay MD m_i to the AP integrated with an MEC server s_j and the energy consumption E_{m_i,s_j}^O for offloading as

$$T_{m_i,s_j}^O = \frac{l_{m_i}^{s_j}}{r\left(P_{m_i,s_j}\right)}, \tag{18}$$

$$E_{m_i,s_j}^O = \frac{P_{m_i,s_j} l_{m_i}^{s_j}}{r\left(P_{m_i,s_j}\right)}. \tag{19}$$

3.3 MEC Server Processing Phase

Since the computing result is usually short, we ignore its feed back time from the MEC server to the local MD. The computing resources of the MEC server are strong enough, so we do not consider the energy consumption for computing at the MEC server. Therefore, the MEC server processing phase consists of computing the directly offloaded segment from the local MD and the indirectly offloaded segment from the relay MD m_i. Let c_s represent the number of CPU cycles used by the MEC server s for each task bit, f_s represent the maximum CPU computing capacity of s, and $f_s^{'}$ represent the computing capacity required by s for the application task. Then the computing delay T_{u,s_k}^C for the directly offloaded segment from the local MD and the computing delay T_{m_i,s_j}^C for the

indirectly offloaded segment form the relay MD at the MEC server s can be described as

$$T^C_{u,s_k} = \frac{c_{s_k} l^{s_k}_u}{f'_{s_k}}, \tag{20}$$

$$T^C_{m_i,s_j} = \frac{c_{s_j} l^{s_j}_{m_i}}{f'_{s_j}}. \tag{21}$$

The delay constraint for the directly offloaded segment on the MEC server includes the offloading delay from the local MD and the computing delay at the MEC server. Then, we have

$$T^O_{u,s_k} + T^C_{u,s_k} \leq T. \tag{22}$$

The delay constraint for the indirectly offloaded segment includes the offloading delay from the local MD to the relay MD and then to the MEC server, as well as the computing delay at the MEC server. Then we get

$$T^O_{u,m_i} + T^O_{m_i,s_j} + T^C_{m_i,s_j} \leq T. \tag{23}$$

Substitute Eq. (7) and Eq. (20) into Eq. (22), and substitute Eq. (10), Eq. (12), Eq. (18) and Eq. (21) into Eq. (23), so we get

$$\frac{l^{s_k}_u}{r\left(P_{u,s_k}\right)} + \frac{c_{s_k} l^{s_k}_u}{f_{s_k}} \leq T, \tag{24}$$

$$\frac{l^{m_i}_u + l^{s_j}_{m_i}}{r\left(P_{u,m_j}\right)} + \frac{l^{s_j}_{m_i}}{r\left(P_{m_i,s_j}\right)} + \frac{c_{s_j} l^{s_j}_{m_i}}{f_{s_j}} \leq T. \tag{25}$$

4 Problem Formulation and Optimal Solution

In this section, we pursue the energy-efficient problem in the process of AP selection and relay selection as well as the task segmentation in an IoT network based on MEC under delay constraint. We also propose and solve the supported maximum task length problem in the scenario.

4.1 Energy Efficient Problem and Optimal Solution

We use $\theta_i = \{0, 1\}$ to represent the single selected offloading relay MD m_i by the local MD, $\theta_j = \{0, 1\}$ to represent the single selected AP s_j by the relay MD, and $\theta_k = \{0, 1\}$ to represent the single selected AP s_k by the local MD for direct offloading, thus we have

$$\sum_{i=1}^{M} \theta_i = 1, \sum_{j=1}^{N} \theta_j = 1, \sum_{k=1}^{N} \theta_k = 1. \tag{26}$$

The process of energy consumption is mainly consist of the following parts: the local MD computing, a relay MD m_i computing, the local MD offloading

to the relay MD m_i, the local MD offloading to the AP s_k, and the relay MD offloading to the AP s_j. Let E represent the total energy consumption generated in the whole process, so we have

$$E = E_u^C + \sum_{i=1}^{M}\sum_{j=1}^{N}\sum_{k=1}^{N}\left(\theta_i\left(E_{u,m_i}^O + E_{m_i}^C\right) + \theta_k E_{u,s_k}^O + \theta_j E_{m,s_j}^O\right). \tag{27}$$

Substitute Eq. (5), Eq. (8), Eq. (10), Eq. (11), Eq. (16) and Eq. (19) into Eq. (27) then we get

$$\begin{aligned} E =& \frac{\gamma_u c_u^3 l_u^3}{T^2} \\ &+ \sum_{i=1}^{M}\sum_{j=1}^{N}\sum_{k=1}^{N}\theta_i\left(\frac{P_{u,m_i}\left(l_u^{m_i}+l_{m_i}^{s_j}\right)}{r\left(P_{u,m_i}\right)} + \frac{r\left(P_{u,s_k}\right)^2\gamma_{m_i}c_{m_i}^3 {l_u^{m_i}}^3}{\left(Tr\left(P_{u,m_i}\right)-\left(l_u^{m_i}+l_{m_i}^{s_j}\right)\right)^2}\right) \\ &+ \theta_k\frac{P_{u,s_k}l_u^{s_k}}{r\left(P_{u,s_k}\right)} + \theta_j\frac{P_{m_i,s_j}l_{m_i}^{s_j}}{r\left(P_{m_i,s_j}\right)}. \end{aligned} \tag{28}$$

By designing a variable of the local MD's task partition vector $\boldsymbol{l} \triangleq [l_u, l_u^{s_k}, l_u^{m_i}, l_{m_i}^{s_j}]$, the energy minimization problem can be expressed as

$$\begin{aligned} (P1): &\min_{\boldsymbol{l},i,j,k} E \\ &\text{s.t. } T \geq 0, \\ &\quad l_u \geq 0, l_u^{s_k} \geq 0, l_u^{m_i} \geq 0, l_{m_i}^{s_j} \geq 0, \\ &\quad \theta_i = \{0,1\}, \theta_j = \{0,1\}, \theta_k = \{0,1\}, \\ &\quad (1), (3), (14), (24), (25) \text{ and } (26). \end{aligned} \tag{29}$$

In problem $(P1)$, $\frac{P_{u,m_i}\left(l_u^{m_i}+l_{m_i}^{s_j}\right)}{r\left(P_{u,m_i}\right)} + \frac{P_{u,s_k}l_u^{s_k}}{r\left(P_{u,s_k}\right)} + \frac{P_{m_i,s_j}l_{m_i}^{s_j}}{r\left(P_{m_i,s_j}\right)}$ is a linear problem. $\frac{{l_u^{m_i}}^3}{\left(l_u^{m_i}+l_{m_i}^{s_j}\right)^2}$ is convex with $l_u^{m_i} \geq 0$ and $l_u^{m_i} + l_{m_i}^{s_j} \geq 0$, then the term $\frac{r\left(P_{u,s_k}\right)^2\gamma_{m_i}c_{m_i}^3{l_u^{m_i}}^3}{\left(Tr\left(P_{u,m_i}\right)-\left(l_u^{m_i}+l_{m_i}^{s_j}\right)\right)^2}$ in problem $(P1)$ is jointly convex as $l_u^{m_i} \geq 0$ and $\frac{l_u^{m_i}+l_{m_i}^{s_j}}{r(P(u,m_i))} < T$. Therefore, it can be concluded that problem $(P1)$ is a convex problem, which can be solved by the convex optimization toolbox.

According to problem $(P1)$, we can get the minimum value of energy consumption based on the cooperative offloading of relay MDs, solve the values of i, j and k to determine the selection of relay MD and the AP and determine each segment length of the application task.

4.2 Supported Maximum Task Length

In the IoT networks, we define the supported maximum task length as the number of data bits supported most by a task under a given delay constraint T. We then formulate the problem as

$$
\begin{aligned}
(P2): &\max_{\boldsymbol{l}} l_{\mathrm{u}} + l_u^{s_k} + l_u^{m_i} + l_{m_i}^{s_j} \\
&\text{s.t. } T \geq 0, \\
&\quad l_u \geq 0, l_u^{s_k} \geq 0, l_u^{m_i} \geq 0, l_{m_i}^{s_j} \geq 0 \\
&\quad (3), (14), (24) \text{ and} (25).
\end{aligned} \tag{30}
$$

Problem $(P2)$ is linear that can be solved effectively through standard convex optimization techniques when m_i, s_j and s_k are confirmed by problem $(P1)$.

5 Numerical Analysis

In this section, we propose two benchmark schemes for comparison with our cooperative offloading scheme through numerical experiments. Our benchmark schemes include

1) Local computing only: The local MD will finish all task computing. From Eq. (3), the maximum application task of the local MD can be described as $\frac{f_u T}{c_u}$. From Eq. (5), the energy consumption of the local MD is $E_u^C = \frac{\gamma_u c_u^3 l_u^3}{T^2}$.

2) Direct offloading: The local MD offloads application tasks directly to the MEC server without the help of the relay MD. Like our proposed cooperative offloading scheme, this scheme maximizes the task length and minimizes the energy consumption by setting $l_u^{m_i} + l_{m_i}^{s_j} = 0$ to solve problem $(P1)$ and problem $(P2)$.

In the numerical experiment, we set a spatial Cartesian coordinate system with a range of $(0 \leq x \leq 500, 0 \leq y \leq 300, 0 \leq z \leq 200)$, and randomly placed a group of relay MDs and a group of AP nodes at different positions in the system. Set the location of the local MD at $(300, 100, 100)$, and randomly place 200 relay MDs and 5 AP nodes in the system.

In order to facilitate the processing of numerical experimental results, we set $\omega_i = \omega_j = \omega_k = 1\,\mathrm{MHz}$, $\sigma_{m_i}^2 = \sigma_{s_j}^2 = \sigma_{s_k}^2 = -70\,\mathrm{dBm}$, $c_u = c_{m_i} = c_{s_j} = c_{s_k} = 10^3\,\mathrm{cycles/bit}$, $P_{u,m_i} = P_{u,s_k} = P_{m_i,s_j} = 40\,\mathrm{dBm}$, $\gamma_u = \gamma_{m_i} = 10^{-26}$, $f_u = f_{m_i} = 1\,\mathrm{GHz}$ and $f_{s_j} = f_{s_k} = 5\,\mathrm{GHz}$. In addition, we assume that the distance from node A to node B is represented by x, and the path loss is $\beta_0 = -60\,\mathrm{db}$ corresponding to the reference distance $x_0 = 10\,\mathrm{m}$, then the path-loss between two MDs is $\beta_0 (x/x_0)^{-\zeta}$, where $\zeta = 3$ is the path-loss exponent [15].

Figure 2 shows the relationship of the supported maximum task length in average versus delay constraint. As the delay constraint increases, so does the maximum length of the corresponding computational task. The supported maximum task length of our proposed scheme is much longer than the other two schemes because the relay MD can help offloading more additional data bits of the task especially for longer delay constraint. Compared with the direct offloading scheme, the local computing only scheme supports less data with the same delay constraint. While the length of the input task exceeds the maximum length, the task will not be completed within the delay constraint, which means that the longer the supported maximum length by the proposed cooperative offloading scheme with the local MD, the greater the computing capacities it has.

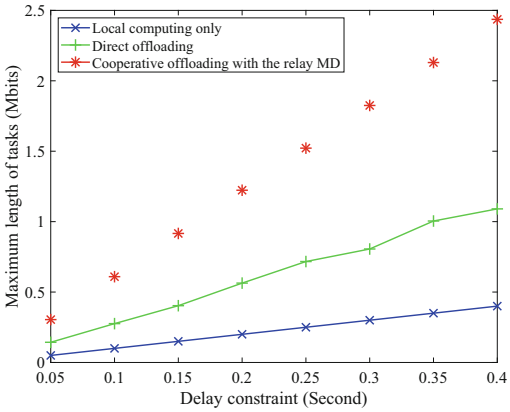

Fig. 2. Supported maximum task length versus the delay constraint T.

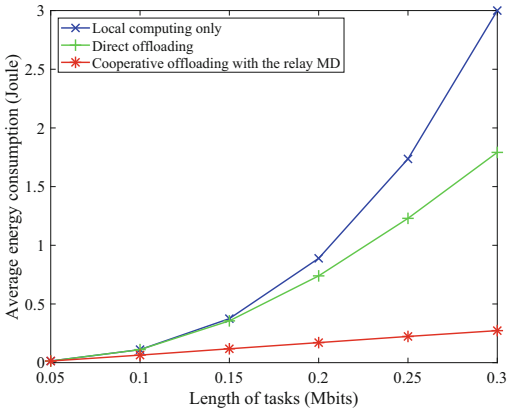

Fig. 3. Average energy consumption under different task length.

Figure 3 shows the relationship between the task length and the average energy consumption under the delay constraint T is set to be $0.3s$. Experimental results show the average energy consumption increases when the application task length becomes longer. By selecting an optimal relay MD, the energy consumption of the proposed scheme significantly outperforms the other two benchmark schemes under different task length. When the input application task is smaller, it will not occupy too much computing resources of the device because tasks will be done at lower CPU frequency, and the performance of the local computing only scheme is similar with other solutions. But the local computing only scheme consumes more energy as the input task length of the application increases, due to the increase of CPU execution frequency. Our proposed scheme consumes relatively much less energy because part of the task can be offloaded to its relay MD and MEC servers for execution.

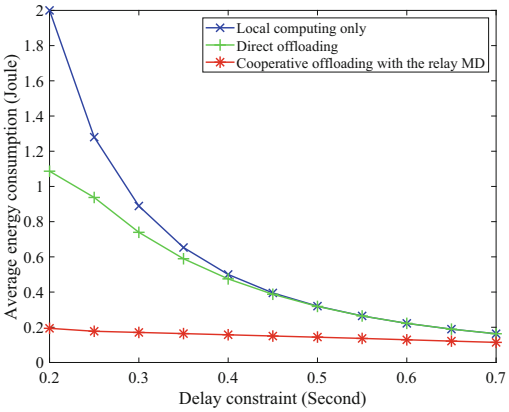

Fig. 4. Average energy consumption under different delay constraints.

Figure 4 shows the relationship between the average minimum energy and the delay constraint when the application task is $L = 0.02$ Mbits. The experimental results show that the energy consumption decreases with the increase of time constraints, which means that the shorter time of the delay constraint, the greater energy consumed by the computation task. Compared with the benchmark schemes, our proposed scheme has the smallest energy consumption at each delay constraint setting, while the local computing only scheme has the largest energy consumption for all tasks are placed on the local MD. However, with the increase of delay constraint, the energy consumption of the three schemes gradually tends to the same scale. When the delay constraint $T = 0.45$ s, the energy consumption of the benchmark schemes gradually becomes the same, because the time constraint becomes loose. Especially when $T = 0.7$ s, the energy consumption of the three schemes differs little, which suggests that our proposed scheme is quite suitable at the condition of strict delay constraint.

6 Conclusions

In this paper, we propose a cooperative energy-efficient offloading scheme under delay constraint for multi-AP MEC in IoT networks. We consider a scenario of a local MD adjacent with many other MDs, one of which can act as the offloading cooperator of the local MD. All the MDs are surrounded by a group of APs, and each AP is deployed an MEC server with abundant computing resources. After building the system model, the computing time, the offloading delay and the energy consumption of local MD processing, relay MD cooperative offloading processing together with the MEC server processing are presented as well. By solving the derived energy-efficient cooperative offloading problem under delay constraint, we can select the most energy-efficient neighbor MD as a relay MD as well as the AP to minimize total energy consumption for task processing. In addition, we formulate and solve a maximum application task length problem

that is supported in MEC-based IoT networks. Experimental results confirm the proposed cooperative offloading scheme under delay constraint achieves better performance than the benchmark schemes with less energy consumption while supports longer application tasks.

References

1. Baidas, M.W.: Offloading-efficiency maximization for mobile edge computing in clustered NOMA networks. In: 2020 11th IEEE Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON), pp. 101–107 (2020)
2. Chen, Y., Zhang, N., Zhang, Y., Chen, X., Wu, W., Shen, X.S.: TOFFEE: task offloading and frequency scaling for energy efficiency of mobile devices in mobile edge computing. IEEE Trans. Cloud Comput., 1 (2019)
3. Fan, W., Liu, Y., Tang, B., Wu, F., Wang, Z.: Computation offloading based on cooperations of mobile edge computing-enabled base stations. IEEE Access **6**, 22622–22633 (2018)
4. Grant, M.: CVX: Matlab software for disciplined convex programming. http://cvxr.com/cvx (2008)
5. Guo, S., Liu, J., Yang, Y., Xiao, B., Li, Z.: Energy-efficient dynamic computation offloading and cooperative task scheduling in mobile cloud computing. IEEE Trans. Mob. Comput. **18**(2), 319–333 (2019)
6. Hu, G., Jia, Y., Chen, Z.: Multi-user computation offloading with D2D for mobile edge computing. In: 2018 IEEE Global Communications Conference (GLOBECOM), pp. 1–6 (2018)
7. Kumar, K., Lu, Y.H.: Cloud computing for mobile users: can offloading computation save energy? Computer **43**(4), 51–56 (2010)
8. Li, M.S., Gao, J., Zhao, L., Shen, X.M.: Deep reinforcement learning for collaborative edge computing in vehicular networks. IEEE Trans. Cogn. Commun. Netw. **6**(4), 1122–1135 (2020)
9. Mao, Y., You, C., Zhang, J., Huang, K., Letaief, K.B.: A survey on mobile edge computing: the communication perspective. IEEE Commun. Surv. Tutorials **19**(4), 2322–2358 (2017)
10. Ning, Z., Dong, P., Kong, X., Xia, F.: A cooperative partial computation offloading scheme for mobile edge computing enabled internet of things. IEEE Internet Things J. **6**(3), 4804–4814 (2019)
11. Niyato, D., Maso, M., Kim, D.I., Xhafa, A., Zorzi, M., Dutta, A.: Practical perspectives on IoT in 5G networks: from theory to industrial challenges and business opportunities. IEEE Commun. Mag. **55**(2), 68–69 (2017)
12. Opadere, J., Liu, Q., Zhang, N., Han, T.: Joint computation and communication resource allocation for energy-efficient mobile edge networks. In: ICC 2019–2019 IEEE International Conference on Communications (ICC), pp. 1–6 (2019)
13. Pan, Y., Chen, M., Yang, Z., Huang, N., Shikh-Bahaei, M.: Energy-efficient NOMA-based mobile edge computing offloading. IEEE Commun. Lett. **23**(2), 310–313 (2019)
14. Saleem, U., Liu, Y., Jangsher, S., Li, Y., Jiang, T.: Mobility-aware joint task scheduling and resource allocation for cooperative mobile edge computing. IEEE Trans. Wireless Commun. **20**(1), 360–374 (2021)

15. Sun, H., Wang, J., Peng, H., Song, L., Qin, M.: Delay constraint energy efficient cooperative offloading in MEC for IoT. In: Gao, H., Wang, X., Iqbal, M., Yin, Y., Yin, J., Gu, N. (eds.) CollaborateCom 2020. LNICST, vol. 349, pp. 671–685. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-67537-0_40
16. Tsai, J.F., Huang, C.H., Lin, M.H.: An optimal task assignment strategy in cloud-fog computing environment. Appl. Sci. **11**(4), 1909–2006 (2021)
17. Wang, S., Guo, Y., Zhang, N., Yang, P., Zhou, A., Shen, X.: Delay-aware microservice coordination in mobile edge computing: a reinforcement learning approach. IEEE Trans. Mob. Comput. **20**(3), 939–951 (2021)
18. Wei, F., Chen, S., Zou, W.: SCADS: simultaneous computing and distribution strategy for task offloading in mobile-edge computing system. China Commun. **15**(11), 149–157 (2018)
19. Xl, A., Liang, Z.B., Ky, C., Ma, D., Yj, E.: A cooperative resource allocation model for IoT applications in mobile edge computing. Comput. Commun. **173**, 183–191 (2021)
20. Zhang, N., Wu, R., Yuan, S., Yuan, C., Chen, D.: RAV: relay aided vectorized secure transmission in physical layer security for internet of things under active attacks. IEEE Internet Things J. **6**(5), 8496–8506 (2019)
21. Zhang, T., Wen, H., Jie, T., Song, H., Xie, F.: Cooperative jamming secure scheme for IWNs random mobile users aided by edge computing intelligent node selection. IEEE Trans. Industr. Inf. **17**(7), 4999–5009 (2021)