Delay Constraint Energy Efficient Cooperative Offloading in MEC for IoT

Haifeng Sun[1(✉)], Jun Wang[2], Haixia Peng[3], Lili Song[1], and Mingwei Qin[4]

[1] School of Computer Science and Technology, Southwest University of Science and Technology, Mianyang 621010, China
dr_hfsun@163.com

[2] Guangdong Key Laboratory of Intelligent Information Processing, Shenzhen University, Shenzhen 518061, China

[3] Department of Electrical and Computer Engineering, University of Waterloo, Waterloo N2L 3G1, Canada

[4] School of Information Engineering, Southwest University of Science and Technology, Mianyang 621010, China

Abstract. Mobile edge computing (MEC) is a promising approach to execute delay-sensitive and computation-intensive applications in the resource-limited IoT mobile devices (IMDs) by offloading computing tasks to MEC servers. In this paper, we propose a neighbor-aided cooperative offloading scheme with delay constraint to improve the energy efficiency in MEC-based Internet of Things (IoT) networks. The network consists of a target IMD with some IMD neighbors, and an access point integrated with an MEC server. The latency-constrained tasks in the IMD can be partially offloaded to and executed by the selected neighboring IMD or the MEC server. Different from other works, our proposed offloading scheme selects the most energy-efficient one from all the neighboring IMDs as the offloading helper. Specifically, we formulate an optimization problem to minimize the total energy consumption while satisfying the computation delay constraint of each task, and obtain the most energy-efficient neighbor with the optimized division of tasks by solving the formulated problem. Moreover, we design an easy neighbor selection scheme with lower time complexity by the weighted value of the transmission rate for each neighbor. Numerical results show that the proposed scheme outperforms benchmark schemes significantly in terms of energy consumption and the supported maximum task length.

Keywords: Cooperative offloading · Delay constraint · Energy efficiency · Mobile edge computing · Neighbor selection

This work was supported in part by Doctoral Scientific Research Foundation of SWUST(16zx7106), Applied Basic Research Programs of Science & Technology Committe Foundation of Sichuan Province (2019YJ0309), Foundation of Sichuan Educational Committee (18ZB0611), and National Key Research & Development Project (2016YFF0104003).

H. Gao et al. (Eds.): CollaborateCom 2020, LNICST 349, pp. 671–685, 2021.
https://doi.org/10.1007/978-3-030-67537-0_40

1 Introduction

Internet of Things (IoT) applications have been growing explosively, which are usually delay-sensitive and computation-intensive, that require high processing capacities within sustainable time constraints [14]. Although IoT mobile devices (IMDs) have been becoming more and more powerful in the central processing unit (CPU), which may still not capable of handling computation tasks within limited time constraints. On the other hand, computation-intensive applications require more energy consumption, which shortens the lifetime of MIDs obviously, and high energy consumption in IMDs poses a significant obstacle for user experience. The traditional solution is to introduce cloud computing by offloading computation tasks to the cloud platform [1], but which imposes tremendous traffic load and induces high latency since data are sent to the cloud platform from plenty of IMDs located in different zones.

Mobile edge computing (MEC), a key technology for the 5th generation (5G) mobile networks, can relieve the shortcoming of centralized cloud computing by pushing the computation capabilities to MEC servers located at network edges, more closer to IMDs, that can provide sufficient computation capacities for IMDs, as well as reduce the traffic burden in core networks [15]. Researchers from academia to industry have been widely promoting MEC technologies to save energy consumption while reduce execution delay of applications in IMDs. Typically, an IMD can offload its computation tasks to an access point (AP) integrated with an MEC server providing rich computation resources.

On the other hand, task offloading is not always beneficial in saving energy of IMDs according to some research results [7]. If the IMD is far away from the AP with worse communication links, some tasks even consume more energy if they were offloaded to the MEC server than processed locally due to the communication overhead in energy. But some works show that the relay node situated between the IMD and the AP can save the communication energy in total [3,8,10].

For a common scenario of an IMD with some neighboring IMD nodes, selecting cooperative nodes for offloading from the neighbors with minimized energy consumption is of great realistic meaning. In some cases, tasks can be partitioned into many different segments for execution, thus many neighbor nodes can be selected as offloading helpers. But in other cases, tasks are indivisible or can only be partitioned into two segments for offloading, then only one neighbor node can be selected as the offloading helper.

In this paper, we consider a scheme of selecting the most energy-efficient cooperative neighbor node as the offloading helper with the execution delay constraint. Helped by the selected neighbor node, the minimized energy consumption is realized among all the neighbor nodes of the IMD. To achieve the goal, we first build a system model to introduce the neighbor-aided cooperative task offloading processes, then formulate the energy consumption and the delay constraint in each process at local, at the cooperative node and at the MEC server, respectively. Afterwards, we derive the optimization problem to minimize the total energy consumption and prove it is convex, so it can be effectively tackled

by conventional methods like toolbox CVX [4]. At last, we perform the numerical results to demonstrate the efficiency of the proposed scheme. In particular, the main contributions of this paper can be summarized as follows.

1) We consider a scenario of a group of randomly located IMDs, and an AP integrated with an MEC server providing rich computation resources. In which an IMD can select one neighbor as the offloading helper due to the very limited number of partitioned segments of its applications.
2) In order to prolong the lifespan of the IMDs, a cooperative scheme of selecting the most energy-efficient neighbor as the offloading helper with execution delay constraint is investigated. The system model and the corresponding optimization problem are presented accordingly.
3) By solving the problem using CVX toolbox, the neighbor with the minimized energy consumption is confirmed and the segment length for task offloading in each process is decided. Extensive numerical experiments validate the advantage of our proposed scheme in neighbor selection and segmentation of tasks for energy saving.

The rest of this paper is organized as follows. In Sect. 2, we review the related work. We specify the system model and problem formulation in Sect. 3. Section 4 describes the proposed energy-efficient problem and then we get the optimal solution. Performance evaluations are illustrated in Sect. 5, and Sect. 6 concludes this paper.

2 Related Work

Saving energy with low execution latency to satisfy the user quality of experience is of great value in IMDs. Executing programs slowly in IMDs can save energy. If the clock speed of the CPU is reduced by half, the execution time doubles, but only one quarter of the energy is consumed [7]. However, it hardly satisfies the delay constraint by many delay-sensitive applications. Sending computation to another machine is not a new idea. For example, cloud computing can save energy for mobile users through computation offloading [13]. But, in the IOT circumstance with tremendous IMDs, the remote centralized cloud server will induce bandwidth congestion and thus longer delay. The idea of MEC moves cloud servers to the network edge, which is smart and effective in dealing with the shortcoming of centralized cloud server schemes, and has been attracting tremendous attention of researchers from academia to industry.

On the other hand, according to various cost/benefit studies, only if the energy consumption for transmitting and receiving data is less than the execution cost in the IMD system locally. Since the communication distance, the wireless channel states, the length of task input-bits, the time constraint and so forth will affect the offloading cost/benefit jointly, energy efficient offloading to optimize the resource allocation is one of the key problems in MEC.

Some research works reveal that, cooperative offloading can save the energy consumption aided by neighbor nodes than offloading the partial task directly

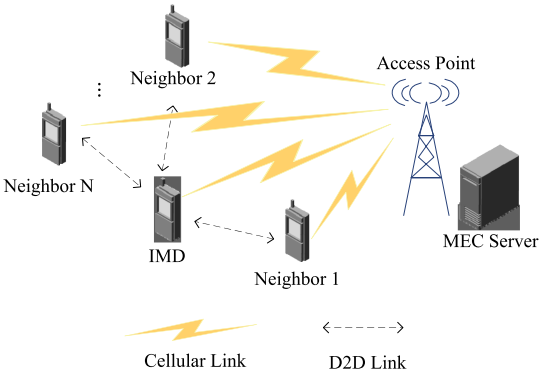

Fig. 1. System model.

to the MEC server especially in bad communication conditions. The authors in [10] study an energy-efficient computation offloading scheme with specialized cooperative nodes, in which N IMDs, M cooperative edge nodes and one cloud server are considered. Tasks in IMDs can be offloaded to one of the edge nodes or forwarded to the cloud server directly decided by the formulated energy optimization problem. Cao *et al.* [3] introduce another scenario of one IMD and an AP with a neighboring node located between them as the helper to assist the computation task offloading with energy efficiency, in which time is divided into four slots for cooperation over the whole block of task computation process. Different from these works, in this paper, we consider a scenario of an IMD surrounded by many neighboring IMD nodes, and the best energy efficient one will be selected as the cooperative offloading helper due to the tasks can only be partitioned into limited segments in the applications.

3 System Model

In this section, we introduce a scenario of a target IMD surrounded by many neighboring IMD nodes in the MEC-based IoT network, then formulate the energy consumption and delay constraint in local processing, neighbor node processing and MEC processing, respectively.

As shown in Fig. 1, we consider a network consisting of an AP integrated with an MEC server providing rich computation resources, one IMD and a set of K neighboring IMDs of it as cooperative candidates for helping offload computation tasks to the MEC server. Since some computation tasks are indivisible or can only be partitioned into a very limited number of segments for execution, we consider the case that only one neighbor node is selected as the offloading helper, achieving the minimized energy consumption with delay constraint. Each segment of the task can be processed locally, offloaded to one of its neighbor nodes through D2D links or to the MEC server directly through cellular links for execution. Further more, segments offloaded to the neighbor node can continually be

offloaded to the MEC server through cellular links. After the task is performed at the neighbor or the MEC server, the result will be sent back to the IMD.

In the system, the D2D links and the cellular links are considered to be deployed over different frequencies, so the D2D links and the cellular links do not interfere with each other [6,12]. Same with many other research works, we suppose the D2D links and the cellular links are working with pro-defined fix bandwidth [9]. We also suppose channels from IMDs to the AP follow the quasi-static block fading. That is, the channel state remains unchanged during the offloading period of one computational segment [5]. In addition, costs of delay and energy consumption for result downloading are not taken into consideration since the result has much less length [11].

We suppose a set of neighbor nodes of the IMD in the network are denoted by $\mathbb{K} = \{1, 2, \ldots, K\}$, and the IMD has the computation tasks of $L > 0$ input-bits with the delay constraint $T \geq 0$.

Consider the case of partial offloading, a task of L input-bits of the IMD can be partitioned into four different segments. Let $l_u \geq 0$, $l_u^s \geq 0$, $l_k \geq 0$ and $l_k^s \geq 0$ denote the number of segment input-bits for local computing in the IMD, offloading to the MEC server from the IMD, offloading to a neighbor node k, $k \in \mathbb{K}$ of the IMD, and offloading to the MEC server from k, respectively. Then we have

$$L = l_u + l_u^s + l_k + l_k^s. \tag{1}$$

Note that the segment of l_k bits executed on the node k, and the other segment of l_k^s bits offloaded from k to the MEC server are both offloaded from the IMD to its neighbor node k. Then, the total offloaded bits from the IMD to its neighbor node k is $l_k + l_k^s$.

The whole processes of task computation and offloading in the system can be summarized into three stages. The first stage is local processing, the second stage is neighbor node processing, and third stage is MEC server processing. In the following section, we will formulate the energy consumption and delay constraint for each stage in detail.

3.1 Local Processing

In the stage of local processing, it includes computation for the partial task of l_u bits at the local IMD, the offloaded partial task of $l_k + l_k^s$ bits to its neighbor node k, and the offloaded partial task of l_u^s bits to the MEC server from the IMD.

Local Computing. Let c_u denote the number of CPU cycles for computing each task input-bit, f_u denote the maximum computation capability (in CPU cycles/s) at the IMD, and f_u' denote the computation capability on demand, respectively. Then the computation latency T_u^C in the IMD is

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

Since the execution of computation tasks is constrained within the delay constraint T, each part of the execution tasks is requested to be finished within T. As a result, the time for executing the computation task with l_u input-bits need to satisfy the requirement $T_u^C \leq T$. Substituting (2) into the requirement and we get $\frac{c_u l_u}{f'_u} \leq T$. Considering $f'_u \leq f_u$, then $\frac{c_u l_u}{f_u} \leq \frac{c_u l_u}{f'_u}$, and thus we get the delay constraint as

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

Let γ_u denote the effective capacitance coefficient of CPU at the IMD [2]. Then the energy consumption E_u^C for local computing at the IMD is [11,16]

$$E_u^C = \gamma_u c_u {f'_u}^2 l_u. \tag{4}$$

From (2) and (4) we get, the longer computation latency, the less energy consumption for local computing. In order to get the minimized energy consumption for the local execution of l_u bits, we set T_u^C with the delay constraint T, that is $T = T_u^C = \frac{c_u l_u}{f'_u}$, and we get $f'_u = \frac{c_u l_u}{T}$, then (4) can be denoted by

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

Computation Offloading to Neighbor Node. Before a neighbor node k computing the offloaded task bits, the IMD will firstly offload $l_k + l_k^s$ task bits to the node k with transmit power $P_{u,k} \geq 0$ through the D2D link. Let $h_{u,k} \geq 0$ denote the channel power gain from the IMD to k, and B_1 the channel bandwidth. Accordingly, the achievable data rate (in bits/s) for task offloading from the IMD to its neighbor node k is

$$r(P_{u,k}) = B_1 \log_2 (1 + \frac{P_{u,k} h_{u,k}}{\sigma_k^2}), \tag{6}$$

where σ_k^2 denotes the noise power at k.

Then the offloading delay $T_{u,k}^O$ and the energy consumption $E_{u,k}^O$ from the IMD to its neighbor node k are given respectively by

$$T_{u,k}^O = \frac{l_k + l_k^s}{r(P_{u,k})}, \tag{7}$$

$$E_{u,k}^O = \frac{P_{u,k}(l_k + l_k^s)}{r(P_{u,k})}. \tag{8}$$

Computation Offloading to MEC. Suppose the partial task offloaded from the IMD to the MEC server through the cellular link with transmit power $P_{u,s} \geq 0$, and $h_{u,s} \geq 0$ denote the channel power gain from the IMD to the MEC server, then the achievable data rate (in bits/s) for task offloading from the IMD to the MEC server is given by

$$r(P_{u,s}) = B_2 \log_2 (1 + \frac{P_{u,s} h_{u,s}}{\sigma_s^2}), \tag{9}$$

where σ_s^2 denotes the noise power at the MEC server and B_2 denotes the cellular channel bandwidth, respectively.

Accordingly, The offloading delay and the energy consumption from the IMD to the MEC server are thus

$$T_{u,s}^O = \frac{l_u^s}{r(P_{u,s})}, \tag{10}$$

$$E_{u,s}^O = \frac{P_{u,s} l_u^s}{r(P_{u,s})}, \tag{11}$$

3.2 Neighbor Node Processing

The stage of neighbor node processing includes computation for the partial task of l_k bits in the neighbor node k and offloading l_k^s bits to the MEC server from k.

Computing at Neighbor Node. Let c_k denote the number of CPU cycles for computing each task input-bit, γ_k denote the CPU effective capacitance coefficient, f_k denote the maximum computation capability (in CPU cycles/s), and f_k' denote the computation capability on demand at the neighbor node k, respectively. Thus, the computing time T_k^C at the neighbor node k is expressed as

$$T_k^C = \frac{c_k l_k}{f_k'}. \tag{12}$$

The delay constraint on the neighbor node includes the transmit delay and the computing delay. Then, we have

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

Substituting (7) and (12) into (13), and because $f_k' \leq f_k$, similarly as in (3), we get the delay constraint as

$$\frac{l_k + l_k^s}{r(P_{u,k})} + \frac{c_k l_k}{f_k} \leq T. \tag{14}$$

Similarly as in (5), the energy consumption E_k^C for local computing at the neighbor node k is

$$E_k^C = \frac{\gamma_k {c_k}^3 {l_k}^3}{(T - T_{u,k}^O)^2}. \tag{15}$$

Substituting (7) into (15) we get

$$E_k^C = \frac{r(P_{u,k})^2 \gamma_k {c_k}^3 {l_k}^3}{(Tr(P_{u,k}) - (l_k + l_k^s))^2} \tag{16}$$

Computation Offloading. Suppose the partial task is offloaded from a neighbor node k through the cellular link with transmit power $P_{k,s} \geq 0$, and $h_{k,s} \geq 0$ denote the channel power gain from the node k to the MEC server. The achievable data rate (in bits/s) for task offloading from k to the MEC server is thus

$$r(P_{k,s}) = B_2 \log_2 (1 + \frac{P_{k,s} h_{k,s}}{\sigma_s^2}), \tag{17}$$

Accordingly, the offloading delay and the energy consumption for task offloading from k to the MEC server can be separately expressed as

$$T_{k,s}^O = \frac{l_k^s}{r(P_{k,s})}, \tag{18}$$

$$E_{k,s}^O = \frac{P_{k,s} l_k^s}{r(P_{k,s})}. \tag{19}$$

3.3 MEC Server Processing

Since the purpose of proposed scheme is to minimize the energy consumption in the MEC-based IoT network and the MEC server has sufficient computation resources in general, the energy consumption of the computation for the offloaded segments at the MEC server is not take into consideration. Therefore, the stage of MEC server processing only includes the offloaded task computing.

We also suppose the partial tasks offloaded to the MEC server from the IMD and its neighbor node do not need to wait for execution in queue. Let c_s denote the number of CPU cycles for computing each task input-bit, γ_s denote the CPU effective capacitance coefficient, f_s denote the maximum computation capability (in CPU cycles/s) at the MEC server, and f_s' denote the computation capability on demand, respectively. Then, we have the time that the MEC server computes the partial tasks offloaded from the IMD and k as

$$T_{u,s}^C = \frac{c_s l_u^s}{f_s'}, \tag{20}$$

$$T_{k,s}^C = \frac{c_s l_k^s}{f_s'}. \tag{21}$$

Similarly as in (3) and (13), the delay constraint at the MEC server includes the offloading delay and the computing delay for tasks offloaded from the IMD and from the neighbor node k, respectively. Then, we have

$$T_{u,s}^O + T_{u,s}^C \leq T. \tag{22}$$

$$T_{u,k}^{O} + T_{k,s}^{O} + T_{k,s}^{C} \leq T. \tag{23}$$

Substituting (10) and (20) into (22), and substituting (7), (18) and (21) into (23), respectively. Thus we get

$$\frac{l_u^s}{r(P_{u,s})} + \frac{c_s l_u^s}{f_s} \leq T, \tag{24}$$

$$\frac{l_k + l_k^s}{r(P_{u,k})} + \frac{l_k^s}{r(P_{k,s})} + \frac{c_s l_k^s}{f_s} \leq T. \tag{25}$$

4 Problem Formulation and Proposed Offloading Approach

In this section, we formulate an energy-efficient problem with delay constraint for the neighbor-aided cooperative MEC-based IoT network, then we get the optimal solution for the problem, and propose an offloading scheme in selecting the cooperative neighbor with minimized energy consumption. We also present a simple weighted value cooperative neighbor selection scheme to reduce the complexity as a comparison.

4.1 Problem Formulation and Optimal Solution

As the MEC server has reliable power supply, we focus on minimize the total energy consumption caused by offloading and computation at IMDs subjected to the task's delay constraint T. The formulation also optimizes the task partition.

We denote the total energy consumption cooperated by a neighbor node k as E_k, then

$$E_k = E_u^C + E_k^C + E_{u,k}^O + E_{u,s}^O + E_{k,s}^O. \tag{26}$$

Substituting (5),(8),(16),(11) and (19) into (26), we get

$$\begin{aligned} E_k = & \frac{\gamma_u {c_u}^3 {l_u}^3}{T^2} + \frac{r(P_{u,k})^2 \gamma_k {c_k}^3 {l_k}^3}{(Tr(P_{u,k}) - (l_k + l_k^s))^2} \\ & + \frac{P_{u,k}(l_k + l_k^s)}{r(P_{u,k})} + \frac{P_{u,s} l_u^s}{r(P_{u,s})} + \frac{P_{k,s} l_k^s}{r(P_{k,s})}. \end{aligned} \tag{27}$$

We design a variable of the IMD's task partition vector $\boldsymbol{l} \triangleq [l_u, l_u^s, l_k, l_k^s]$. The delay constraint energy minimization problem is then formulated as

$$(\text{P1}): \quad \min_{\boldsymbol{l}} E_k \tag{28a}$$

$$\text{s.t.} \quad T \geq 0, \tag{28b}$$

$$l_u \geq 0, l_u^s \geq 0, l_k \geq 0, l_k^s \geq 0, \tag{28c}$$

$$(1), (3), (14), (24) \text{ and } (25),$$

where (28c) is the constant of delay constraint required by tasks at the IMD. Since $\frac{{l_k}^3}{(l_k+l_k^s))^2}$ is convex with $l_k \geq 0$ and $l_k + l_k^s > 0$, then the term $\frac{r(P_{u,k})^2 \gamma_k {c_k}^3 {l_k}^3}{(Tr(P_{u,k})-(l_k+l_k^s))^2}$ in the objective function is jointly convex with respect to $l_k \geq 0$ and $\frac{l_k+l_k^s}{r(P_{u,k})} < T$. So we get (P1) is convex. Thus (P1) can be optimally solved by the using convex optimization toolbox CVX [4].

By solving problem (P1), we get the minimized energy consumption of each neighbor node for the neighbor-aided cooperative offloading progress, and then we can get one neighbor node with the minimized energy consumption among them. But as we know, the computation complexity is high to solve the convex optimization problem. We hereby design a weighted value cooperative neighbor selection scheme.

4.2 Weighted Value Cooperative Neighbor Selection Scheme

For neighbor-aided cooperative offloading, the energy consumption for computation in each neighbor differs little since the computing power of IMDs are similar for many cases, while the main difference for energy consumption is in the process of offloading, so we can only consider the energy consumption for the offloading process. From (8) and (19) we conclude that the offloading energy consumption is proportional to the transmit power and the length of offloading bits, while inversely proportional to the data rate. Suppose the IMDs have identical transmit power, then we can define the weighted value cooperative neighbor selection scheme as

$$k^* = \arg\min_{k \in \mathbb{K}} (\alpha r(P_{u,k}) + (1-\alpha) r(P_{k,s})), 0 \leq \alpha \leq 1, \tag{29}$$

where α is the coefficient to evaluate the importance between the communication rates from the IMD to its neighbor node k, and from the neighbor node k to the MEC server. The time complexity of the scheme is $O(n)$.

Apparently, the vector $\boldsymbol{l}$ and energy consumption E^* of the neighbor node k^* confirmed by (29) can also be got by solving problem (P1).

4.3 Maximum Task Length

The maximum task length discloses how many bits of a task are supported within the given delay constraint T in the MEC-based IoT network. The maximum length of the task input-bits for the neighbor node k of the IMD will be derived when the task is executed at local, at the neighbor node and at the MEC server, simultaneously. Because in this case, the three nodes can fully take advantage

of their available communication and computation resources. We then formulate the problem as

$$\begin{aligned} (\text{P2}): \quad & \max_{\boldsymbol{l}} \quad l_u + l_u^s + l_k + l_k^s && \text{(30a)} \\ \text{s.t.} \quad & (3), (14), (24), (25), (28\text{b}) \text{ and } (28\text{c}) \end{aligned}$$

Note problem (P2) is a linear program and can thus be efficiently solved for every neighbor node k via standard concave optimization techniques, and hereby we can get a neighbor with the maximum task length. The maximum length of the task input-bits for the neighbor k^* confirmed by (29) can also be got.

5 Numerical Results

In this section, we provide the numerical results to evaluate the performance of the proposed two neighbor-aided cooperative offloading schemes compared with two benchmark schemes. The proposed offloading schemes include

1) Offloading with the best neighbor: The cooperative neighbor with the minimized energy consumption and the maximum task length in the system among all the neighbor nodes of the IMD are selected by solving problem (P1) and problem (P2) for each node, separately.
2) Offloading with the neighbor of weighted value: The cooperative neighbor k^* is selected by solving (29), and the minimized energy consumption as well as the maximum task length in the system are got by solving problem (P1) and problem (P2) by setting $k = k^*$, separately.

The benchmark schemes are

1) Local computing: The IMD executes whole computation tasks locally by itself. We can get the maximum length of task input-bits by solving $\frac{Tf_u}{c_u}$ from (3), and get the energy consumption $E_u^C = \gamma_u c_u f_u^2 L$ by Eq. (5).
2) Offloading without neighbor: The IMD offloads computation tasks to the MEC server without corporation of a neighbor. Like the scheme of offloading with the best neighbor, this scheme corresponds to solving problem (P1) and problem (P2) by setting $l_k + l_k^s = 0$ to get the minimized energy consumption and the maximum task length, separately.

In the simulation set-up, we consider the neighbors of the IMD are randomly located within the area of a rectangular coordinate system in order to get the influence of the different locations of the neighbor nodes. Let (c_k^x, c_k^y) denote the location coordinates in X-axis and Y-axis of a node k in the rectangular coordinate system. Then we set $0 \leq c_k^x \leq 500m$ and $0 \leq c_k^y \leq 200m$. Therefore, the whole rectangular area is evenly divided into a grid network of $26*11$ with the IMD located at $(100, 0)$ and the AP at $(400, 0)$, separately. At last, we randomly

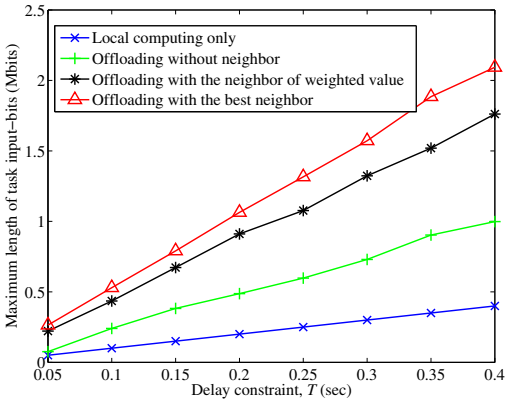

Fig. 2. Average maximum length of task input-bits versus the delay constraint T.

set 20 neighbor nodes located at different intersections in the whole rectangular areas.

Let d denote the distance from the transmitter to the receiver, then the path-loss between any two nodes is $\beta_0(d/d_0)^{-\xi}$, where $\beta_0 = -60$ db corresponds to the path-loss at the reference distance of $d_0 = 10$ m, and $\xi = 3$ is the path-loss exponent. Furthermore, we set $B_1 = B_2 = 1$ MHz, $\sigma_k^2 = \sigma_s^2 = -70$ dBm, $c_u = c_k = c_s = 10^3$ cycles/bit, $P_{u,k} = P_{u,s} = P_{k,s} = 40$ dBm [3], $\gamma_u = \gamma_k = 10^{-26}$, $f_u = f_k = 1$ GHz and $f_s = 5$ GHz.

Figure (2) shows the average maximum length of task input-bits (Mbits) versus the delay constraint T. It is observed that the maximum length of task input-bits increases as the delay constraint becomes longer. In detail, the scheme of offloading with the best neighbor supports the maximum length of task input-bits compared with other schemes. The scheme of the neighbor selected by the weighted value is better than the other two benchmark schemes of local computing and offloading without neighbor aiding. The reason is the adopted offloading strategy can assist the IMD by executing more extra parts of the tasks within the delay constraint. Compared the scheme of local computing only with the scheme of offloading without neighbor, the latter one can execute more data within the delay constraint. Tasks can not be finished within the delay constraint when the length of task input-bits exceeds the supported maximum length, which means that the proposed scheme of offloading with the best neighbor supports the longer maximum length as well as the bigger computation capacities.

Figure (3) shows the average minimum energy consumption versus the delay constraint T with $L = 0.02$ Mbits. The results demonstrate that the minimum energy consumption decreases with the prolonged delay constraint. The proposed scheme of offloading with the best neighbor consumes the least energy for all cases, while the scheme of local computing only consumes the most energy especially when the delay constraint is smaller. The differences between schemes become smaller as the delay constraint becomes longer, and eventually almost

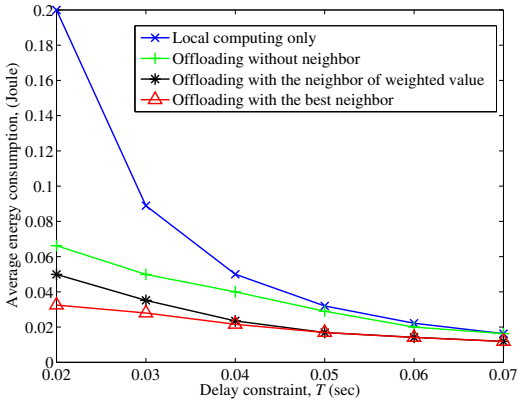

Fig. 3. Average minimum energy consumption versus delay constraint T.

keep in with the same scale, due to more tasks will be executed locally for the slack long delay constraint.

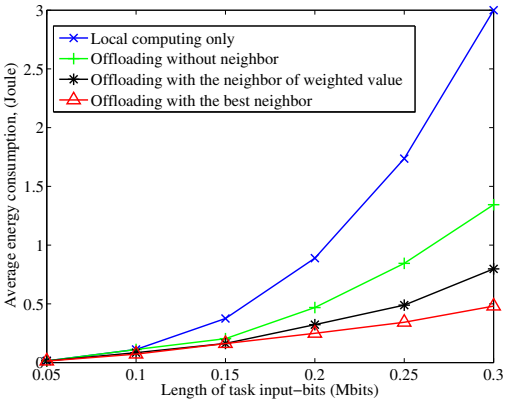

Fig. 4. Minimum energy consumption versus the length of task input-bits (Mbits).

Figure (4) shows the minimum energy consumption versus the length of task input-bits (Mbits) with the delay constraint $T = 0.3s$. The simulation results demonstrate that the minimum energy consumption grows exponentially with the prolonged length of task input-bits, and the proposed scheme of offloading with the best neighbor out performs the others at all cases with lower energy consumption. When the task length is small, the scheme of local computing achieves the similar performance compared with others due to the task can be finished with lower CPU execution frequency within the delay constraint. But when the task length increases, local computing will cost more energy for the

increased CPU execution frequency, while the proposed scheme can offload some parts of the tasks to its neighbor and the MEC server for execution.

6 Conclusion

In this paper, we have investigated how to select the most efficient neighbor for cooperative offloading with delay constraint in a common scenario of a target IMD neighbored with some IMDs in the MEC-based IoT network. After building the system model, we formulate the energy consumption and delay constraint problems in local processing, neighbor node processing and MEC processing, respectively. Hereby, an energy-efficient problem with delay constraint for the neighbor-aided cooperative IoT edge network is formulated. By solving the optimization problem, the neighbor of the target IMD with the least energy consumption is selected. We also present a simple weighted value cooperative neighbor selection scheme to reduce the complexity as a comparison. Moreover, in order to get the supported maximum task length in the MEC-based IoT network, we formulate the task length optimization problem. Numerical results indicate that the proposed scheme outperforms the benchmark schemes significantly in terms of the supported maximum task length and the energy efficiency, which is feasible and effective in the MEC-based IoT. In the future, we will research the scheme of selecting a set of neighbors with energy efficiency and execution delay constraint.

References

1. Barrameda, J., Samaan, N.: A novel statistical cost model and an algorithm for efficient application offloading to clouds. IEEE Trans. Cloud Comput. **6**(3), 598–611 (2018)
2. Burd, T.D., Brodersen, R.W.: Processor design for portable systems. J. VLSI Sig. Process. Syst. Sig. Image Video Technol. **13**(2), 203–221 (1996)
3. Cao, X., Wang, F., Xu, J., Zhang, R., Cui, S.: Joint computation and communication cooperation for energy-efficient mobile edge computing. IEEE Internet Things J. **6**(3), 4188–4200 (2019)
4. Grant, M., Boyd, S.: CVX: Matlab software for disciplined convex programming, version 2.1 (2014). http://cvxr.com/cvx
5. Guo, S.T., Liu, J.D., Yang, Y.Y., Xiao, B., Li, Z.T.: 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 **4**, 51–56 (2010)
8. 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)

9. Peng, H., Ye, Q., Shen, X.: Spectrum management for multi-access edge computing in autonomous vehicular networks. IEEE Trans. Intell. Transp. Syst. 1–12 (2019)
10. Vu, T.T., Huynh, N.V., Hoang, D.T., Nguyen, D.N., Dutkiewicz, E.: Offloading energy efficiency with delay constraint for cooperative mobile edge computing networks. In: IEEE Global Communications Conference. IEEE, New York (2018)
11. Wang, J., Feng, D., Zhang, S., Tang, J., Quek, T.Q.S.: Computation offloading for mobile edge computing enabled vehicular networks. IEEE Access **7**, 62624–62632 (2019)
12. Wu, Y., Chen, J.C., Qian, L.P., Huang, J.W., Shen, X.: Energy-aware cooperative traffic offloading via device-to-device cooperations: an analytical approach. IEEE Trans. Mob. Comput. **16**(1), 97–114 (2017)
13. Yang, K., Ou, S., Chen, H.H.: On effective offloading services for resource-constrained mobile devices running heavier mobile Internet applications. IEEE Commun. Mag. **46**(1), 56–63 (2008)
14. Zhang, N., et al.: Physical layer authentication for internet of things via WFRFT-based Gaussian tag embedding. IEEE Internet Things J. https://doi.org/10.1109/JIOT20203001597
15. 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)
16. Zhang, W., Wen, Y., Guan, K., Kilper, D., Luo, H., Wu, D.O.: Energy-optimal mobile cloud computing under stochastic wireless channel. IEEE Trans. Wireless Commun. **12**(9), 4569–4581 (2013)