Energy Minimization for D2D-Assisted Mobile Edge Computing Networks

Yuan Kai*, Junyuan Wang† and Huiling Zhu*
*School of Engineering and Digital Arts, University of Kent, United Kingdom
E-mail: {yk69, h.zhu}@kent.ac.uk
†Department of Computer Science, Edge Hill University, Ormskirk, United Kingdom
E-mail: junyuan.wang@edgehill.ac.uk

***Abstract*—This paper addresses the energy minimization problem in Device-to-Device (D2D) assisted Mobile Edge Computing (MEC) networks under the latency constraint of each individual task and the computing resource constraint of each computing entity. The energy minimization problem is formed as a two-stage optimization problem. Specifically, in the first stage, an initial feasibility problem is formed to maximize the number of executed tasks and the global energy minimization problem is tackled in the second stage while maintaining the maximum number of executed tasks. Both of the optimization problems in two stages are NP-hard, therefore a low-complexity algorithm is developed for the initial feasibility problem with a supplementary algorithm further proposed for energy minimization. Simulation results demonstrate the near-optimal performance of the proposed algorithms and the fact that with the assistance of D2D communication, the number of executed tasks is greatly increased and the energy consumption per executed task is significantly reduced in MEC networks, especially in dense user scenario.**

***Index Terms*—Mobile edge computing, device-to-device communication, energy minimization, task offloading.**

I. Introduction

With the increasing popularity of user equipments (UEs) such as smartphones and hand-held devices, more and more resource-hungry applications like high definition video streaming, real-time online gaming and virtual reality applications are developing rapidly. However, vast computation-intensive and latency-critical applications/tasks bring severe challenges to the physical limitations of devices, such as CPU capability and battery life.

In recent years, a new paradigm called Mobile Edge Computing (MEC) is envisioned as a promising technology for releasing the idle computation resource at the wireless network edge infrastructures, i.e. base stations and access points (APs), instead of centralized cloud centre [1]. Although the energy and time consumption can be greatly reduced by eliminating the transmission distance from edge infrastructures to cloud centre, the energy and time consumption for the wireless communication and task computing still remains, which needs to be managed carefully.

With respect to the energy consumption and latency requirement of each individual task generated by UEs, most of the previous research focused on the resource management of the infrastructure-based MEC networks, where the tasks generated by UEs can be offloaded to the edge infrastructures, by considering the disciplines of both wireless communications and mobile computing [2]–[5]. However, there exist two limitations in such MEC networks: 1) Despite the relatively high computing power at the edge infrastructures compared to each device, it has to be shared by a large number of tasks, hence reducing the gap in the computation latency; 2) Since the UEs are spatially distributed in the cellular network, the distance between UEs and infrastructures could be very long, especially for the cell edge UEs, resulting unacceptable task offloading latency. Hence, some of the latency-critical tasks may not be able to accomplished. In other words, there may exist non-negligible number of infeasible tasks in MEC networks [5].

To remedy the limitations of MEC network, user-cooperative computing have draw much attention recently [6]–[8]. In [6], a self-coordinated protocol for user-cooperative computing system was firstly proposed. [7] investigated the adaptive offloading control at the UE side based on the prediction of available computing power. An online distributed fog network was proposed in [8] to minimize the maximum computational latency. However, the user-oriented protocols may result in low stability, reliability and efficiency. Moreover, there is no commitment for cooperation without central control. Hence, based on game theory, [9] investigated the price for the task execution to incentivizing the UEs for collaboration.

In view of the above prior works, scattered computing power in massive UEs can be utilized to fulfil the ever-increasing computation demand from the aspect of mobile computing. Also, one key fact that is overlooked is that the offloading distance can be significantly reduced by user cooperative computing. From the wireless communication perspective [11]–[20], short transmit distance leads to high data rate communication with low power consumption and low delay, which can diminish the task offloading latency and energy consumption. In addition, substantial multi-user diversity gains can also be achieved with massive potential cooperative UEs. Therefore, radio resource management for user cooperative computing networks is vital to reduce the number of infeasible tasks as well as minimize the total energy consumption, which was unfortunately not addressed properly in previous research works.

978-1-5386-8088-9/19/$31.00 ©2019 IEEE

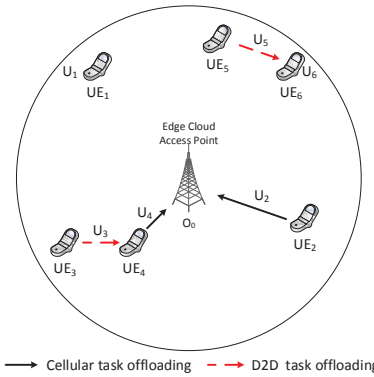

Fig. 1. D2D-Assisted Mobile Edge Computing Network.

As Device-to-Device (D2D) communication was proposed to enable the direct communication between two nearby UEs [11], [21], D2D-assisted MEC is considered in this paper to allow the task offloading between UEs in addition to the network infrastructures. Joint radio-and-computation resource management for D2D-assisted MEC networks is investigated in two stages. Specifically, in the first stage, a low-complexity algorithm is developed to maximize the number of executed tasks subject to the latency constraint of each individual task and the computing resource constraint of each computing entity, i.e. edge AP and UEs. A supplementary algorithm is then proposed in the second stage to minimize the total energy consumption while maintaining the maximum number of executed tasks. Simulation results show that the number of executed tasks and the corresponding energy consumption per executed task with our proposed algorithms are very close to those with the optimal exhaustive search. Furthermore, it is revealed that the number of executed tasks in MEC networks is significantly increased with the assistance of D2D communication, while the energy consumption per executed task is greatly reduced, especially in dense UE scenarios.

The remainder of this paper is organized as follows. Section II describes the system model of D2D-assisted MEC network. Section III presents the feasibility problem and proposes a low-complexity task allocation algorithm. The energy minimization problem is then studied with a supplementary algorithm developed in section IV. Simulation results and discussions are given in Section V, followed by the conclusions in Section VI.

II. System Model

As shown in Fig. 1, we consider that there are N active UEs, denoted by $\mathcal{N} = \{1, 2, ..., N\}$, each of which has one computation-intensive task to be executed. Let $\mathcal{U} = \{U_1, \cdots, U_i, \cdots U_N\}$ denotes the set of tasks to be executed at one time, where task U_i is assumed to be generated at UE i. For any task $U_i \in \mathcal{U}$, let F_i, D_i, $T_{i,max}$ denote the total number of the CPU cycles to be computed, the data size of offloading package and the corresponding time constraint of task U_i, respectively. D_i and F_i can be obtained by using the approaches provided in [22].

It is assumed that device-to-device (D2D) communication is available between any two UEs in the cell. As a result, each task can be offloaded either to the edge access point (AP) located at the centre of the cell or to any other UE via D2D communication. The set of possible computing entities, including the edge AP and all the UEs that each task can be allocated to, is defined as $\mathcal{M} = \{0, \mathcal{N}\}$, with $|\mathcal{M}| = M$, where 0 is the index of the edge AP. $a_{i,j}$, $i \in \mathcal{N}$, $j \in \mathcal{M}$ is defined as the task allocation indicator to show that whether the task from the UE i is allocated to the computing entity j. Thus, one has

$$a_{i,j} \in \{0, 1\}, \forall i \in \mathcal{N}, \forall j \in \mathcal{M}, \tag{1}$$

where $a_{i,j} = 1$ denotes that the UE i decides to allocate its task U_i to the computing entity j and 0 otherwise. As each task can only be executed at one computing entity or may not be able to be completed anywhere due to lack of communication or computation resources, it can be easily obtained that

$$\sum_{j \in \mathcal{M}} a_{i,j} \leq 1, \forall i \in \mathcal{N}. \tag{2}$$

A. Energy Consumption for Task Execution

In this paper, it is considered that the computing resource of computing entity j is denoted as f_j, and multiple tasks can be executed at one computing entity and its computing resource is shared by the tasks allocated to it. That is,

$$\sum_{i \in \mathcal{N}} a_{i,j} \cdot f_{i,j} \leq f_j, \forall j \in \mathcal{M}, \tag{3}$$

where $f_{i,j}$ denotes the allocated computing resource for task U_i at computing entity j.

The execution time of task U_i at computing entity j, $T^C_{i,j}$, is then obtained as

$$T^C_{i,j} = \frac{F_i}{f_{i,j}}, \quad \forall i \in \mathcal{N}, \forall j \in \mathcal{M}, \tag{4}$$

and the corresponding computation energy consumption, $E^C_{i,j}$, is give by [23]

$$E^C_{i,j} = T^C_{i,j} \cdot p^C_{i,j} = \kappa_j (f_{i,j})^{\nu_j - 1} F_i, \quad \forall i \in \mathcal{N}, \forall j \in \mathcal{M}, \tag{5}$$

where $p^C_{i,j} = \kappa_j (f_{i,j})^{\nu_j}$ is the execution power of task U_i with allocated computing resource $f_{i,j}$. $\kappa_i > 0$ is the effective switched capacitance and $\nu_j \geq 2$ is the positive constant.

B. Energy Consumption for Task Offloading

Note that task U_i is not offloaded if $a_{i,i} = 1, \forall i \in \mathcal{N}$, i.e. UE i executes its own task. If UE i decides to offload its task U_i to computing entity j where $j \in \mathcal{M} \setminus i$, the task data D_i has to be transmitted to the destination computing entity j. By assuming that the transmit power of each device is fixed to p^{Tr}, the data rate, $r_{i,j}$, can be obtained as

$$r_{i,j} = B \log_2 \left(1 + \frac{p^{Tr} \cdot |g_{i,j}|^2}{\sigma^2} \right), \quad \forall i \in \mathcal{N}, \forall j \in \mathcal{M} \setminus i, \tag{6}$$

where $g_{i,j}$ is the channel coefficient from UE i to computing entity j, and $|g_{i,j}|^2 = {d_{i,j}}^{-\alpha} \cdot |h_{i,j}|^2$. Here, $h_{i,j}$ is small-scale fading with $h_{i,j} \sim \mathcal{CN}(0, 1)$, $d_{i,j}$ is the distance between UE i and computing entity j and α is the path-loss exponent. σ^2

denotes the additive white Gaussian noise power. Note that dedicated spectrum with fixed bandwidth, B, is assumed to be allocated to each UE for task offloading. The consumed transmission time of task U_i to computing entity j is given by

$$T_{i,j}^{Tr} = \begin{cases} \frac{D_i}{r_{i,j}} & \forall i \in \mathcal{N}, j \in \mathcal{M} \setminus i; \\ 0 & \forall i \in \mathcal{N}, i = j. \end{cases} \tag{7}$$

The transmission energy consumption of offloading task U_i can be then obtained as

$$E_{i,j}^{Tr} = p^{Tr} \cdot T_{i,j}^{Tr}, \ \ \forall i \in \mathcal{N}, j \in \mathcal{M}. \tag{8}$$

The data size of the computation results are considered to be much smaller than the offloading data size, hence the time and energy consumption of sending the results back are ignored in this paper.

The energy consumption of each executing task consists of computation energy and offloading energy. Thus, the total energy consumption can be expressed as

$$E_{tot} = \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{M}} a_{i,j} \left(E_{i,j}^{Tr} + E_{i,j}^{C} \right). \tag{9}$$

It is obvious that if all the tasks in $\mathcal{U}$ are dropped, i.e. $\sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{M}} a_{i,j} = 0$, the minimum energy consumption can be achieved, with $E_{tot} = 0$, and the problem becomes trivial. Moreover, due to the limited computing resource of each entity, one latency-critical task may not be successfully completed under the D2D-assisted MEC network. Therefore, in this paper, we focus on the energy minimization problem while achieving the maximum number of feasible tasks under D2D-assisted MEC networks. For the sake of clarity, a two-stage optimization problem is formulated, which contains the initial feasibility problem and the energy minimization problem.

III. Initial Feasibility Problem

A. Problem Formulation and Analysis

In the first stage, we aim to maximize the number of tasks which can be successfully completed, either through offloading to other UEs, the edge AP or executing themselves. The initial feasibility optimization problem can be formulated as

$$\begin{aligned} \mathcal{P}1: \ & \max_{\{a_{i,j}\},\{f_{i,j}\}} \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{M}} a_{i,j} \\ & \text{Subject to}: (1), (2), (3); \\ & \sum_{j \in \mathcal{M}} a_{i,j} \left(T_{i,j}^{C} + T_{i,j}^{Tr} \right) \leq T_{i,max}, \forall i \in \mathcal{N}, \end{aligned} \tag{10}$$

where (10) follows the constraint that each task U_i is delay-sensitive and needs to be finished within $T_{i,max}$.

Since (2) indicates that each task can be assigned to one computing entity at most, the time constraint of each task can be replaced by the computing resource requirement of each task at each computing entity, $f_{i,j}$, by substituting (4) and (7) into (10), which is

$$f_{i,j} \geq f_{i,j}^{min} = \frac{F_i}{T_{i,max} - D_i / r_{i,j}}, \tag{11}$$

where $f_{i,j}^{min}$ denotes the minimum computing resource requirement of task U_i at computing entity j.

It can be easily seen that the equality of (11) holds for the optimal solution sets $\{a_{i,j}^*\}$ and $\{f_{i,j}^*\}$ of $\mathcal{P}1$. Hence, $a_{i,j}^*$ and $f_{i,j}^*$ must satisfy

$$f_{i,j}^* = a_{i,j}^* \cdot f_{i,j}^{min}. \tag{12}$$

Since it is not possible to allocate negative computing resource, $f_{i,j}^*$ needs to be positive value. That is, $a_{i,j}^* = 0$ when $f_{i,j}^{min} \leq 0$. Therefore, by combining (3) and (12), $\mathcal{P}1$ can be reformulated as

$$\begin{aligned} \mathcal{P}2: \ & \max_{\{a_{i,j}\}} \sum_{i \in \mathcal{N}} \sum_{j \in \mathcal{M}} a_{i,j} \\ & \text{Subject to}: (1), (2); \\ & \sum_{i \in \mathcal{N}} f_{i,j}^{min} \cdot a_{i,j} \leq f_j, \forall j \in \mathcal{M}; \qquad (13) \\ & a_{i,j} = 0, \text{if } f_{i,j}^{min} \leq 0, \forall i \in \mathcal{N}, j \in \mathcal{M}. \qquad (14) \end{aligned}$$

The optimization problem $\mathcal{P}2$ reduces to a multiple knapsack problem, when $f_{i,j}^{min}$ is the same for task U_i at all the computing entities. As the multiple knapsack problem is NP-hard and $\mathcal{P}2$ is at least as hard as the multiple knapsack problem, there is no computational efficient approach to solve $\mathcal{P}2$ optimally. Therefore, a low-complexity task allocation algorithm is developed in next subsection.

B. Algorithm Design

As it can be seen in (11), the offloading rate $r_{i,j}$ for task U_i to computing entity j would lead to different minimum computing resource requirement, $f_{i,j}^{min}$, and the offloading rate $r_{i,j}$ is only related to the offloading channel between computing entity i and j, $g_{i,j}$. Since UEs are assumed to be independent and identically distributed (i.i.d.) within the cell and dedicated spectrum is allocated for each possible offloading transmission, the offloading rates are independent with each other. That is, $f_{i,j}^{min}$ will not be affected by the task allocation result. However, each computing entity has limited amount of avaliable computing resource, f_j, as shown in (13). To illustrate the effect of (13) and (14), we therefore define the auxiliary weight variable of task U_i at computing entity j as $\Delta f_{i,j}$, shown as

$$\Delta f_{i,j} = \begin{cases} \frac{f_{i,j}^{min}}{f_j'} & \text{if } f_{i,j}^{min} > 0; \\ \infty & \text{otherwise}, \end{cases} \tag{15}$$

where f_j' denotes the current avaliable computing resource at entity j.

Based on the defined auxiliary variable $\Delta f_{i,j}$, an initial task allocation (ITA) algorithm is proposed to solve $\mathcal{P}2$. The main idea is to allocate N tasks through N iterations, as each task can only be allocated to one computing entity. During iteration k, task U_{i^*} is allocated to computing entity j^* and $a_{i^*,j^*} = 1$, where $(i^*, j^*) = \arg\min[\Delta f_{i,j}]_{(N-k+1)\times M}$ and $[\Delta f_{i,j}]_{(N-k+1)\times M}$ denotes the auxiliary weight variable matrix for the remaining $N - k + 1$ tasks at M computing

Algorithm 1 Initial Task Allocation (ITA) algorithm

Initialization: Set the task indicator k=1, the task allocation indicator matrix $[a_{i,j}]_{N\times M} = \mathbf{0}$, calculate $[f_{i,j}^{min}]_{N\times M}$ according to (11), $[\Delta f_{i,j}]_{N\times M}^{(0)}$ according to (15).
if $k < N$ and $\min[\Delta f_{i,j}]_{(N-k)\times M}^{(k)} < 1$ **then**
  Based on $[\Delta f_{i,j}]_{(N-k+1)\times M}^{(k-1)}$, calculate $\mathcal{U}_q^{(k-1)}$.
  if $\mathcal{U}_q^{(k-1)} \neq \emptyset$ **then**
    update $[\Delta f_{q,j}]_{|\mathcal{U}_q|\times M}^{(k)}$,then search for $(q^*, j^*) = \arg\min[\Delta f_{q,j}]_{|\mathcal{U}_q|\times M}^{(k)}$ and set $a_{q^*,j^*} \leftarrow 1$
  else
    according to $[\Delta f_{i,j}]_{(N-k+1)\times M}^{(k-1)}$, search for $(i^*, j^*) = \arg\min[\Delta f_{i,j}]_{(N-k+1)\times M}^{(k-1)}$ and update $[a_{i,j}^*]_{N\times M}$.
  end if
  Update $[\Delta f_{i,j}]_{(N-k)\times M}^{(k)}$ according to (15) and (16).
else
  Output $[a_{i,j}^{(p1)}]_{N\times M} = [a_{i,j}^*]_{N\times M}$.
end if

entities. After iteration k, $\{\Delta f_{i,j^*} : \forall i \in \mathcal{N} \setminus i^*\}$ needs to be updated according to (15), where

$$f'_{j^*} = f_{j^*} - \sum_{i\in\mathcal{N}} f_{i,j^*}^{min} \cdot a_{i,j^*}. \tag{16}$$

Note that during the task allocation process, there might exist some special tasks, each of which has only one feasible entity, so called *exclusive* tasks. Specifically, as the mobile UEs are randomly distributed in the system, the number of available computing entity for each task is different due to its location and the individual time constraint (10). An *exclusive* task can only be processed at one idle entity, which fulfils its time constraint. Therefore, to maximize the number of feasible tasks, we propose to give priority to the *exclusive* task set, denoted by $\mathcal{U}_q \subseteq \mathcal{U}$. That is, if $\mathcal{U}_q \neq \emptyset$ at the beginning of iteration k, then one of the *exclusive* tasks U_{q^*} needs to be allocated to entity j^* at iteration k and $a_{q^*,j^*} = 1$, where $(q^*, j^*) = \arg\min[\Delta f_{q,j}]_{|\mathcal{U}_q|\times M}$.

It can be easily seen that after N iterations at most, the ITA algorithm converges, and the task allocation matrix $[a_{i,j}^{(p1)}]_{N\times M}$ can be obtained and the ITA algorithm is summarized in Algorithm 1.

IV. Energy Minimization Problem

In this section, we focus on investigating the energy minimization problem while maintaining the maximum number of feasible tasks achieved by solving the initial feasibility problem.

A. Problem Formulation

As discussed in section III, the minimum computing resource requirement of task U_i at computing entity j, $f_{i,j}^{min}$, can be obtained from (11) based on its individual time constraint. Note that the offloading energy consumption, $E_{i,j}^{Tr}$, is determined by the task allocation, and the computation energy

Algorithm 2 Iterative Switching Task (IST) algorithm

Initialization: Outer iteration number $ot = 1$. Set the task allocation indicator matrix $[a_{i,j}^{(1)}]_{N\times M} = [a_{i,j}^{(p1)}]_{N\times M}$ and calculate $[f_{i,j}^{min}]_{N\times M}$ according to (11).
repeat
  Set task iterative number $k = 1$, number of switching operations $N_s = 0$, and the maximum number of iterations as $\left|\mathcal{U}_F^{(p1)}\right|$.
  repeat
    Calculate the residual computing resource for each entity $j \in \mathcal{M}$, $\{f_j^{Re}\}^{(k-1)}$. For $U_k \in \mathcal{U}_F^{(p1)}$, obtain j_k, where $a_{k,j_k}^{(ot)} = 1$.
    if $f_{k,j_k}^{min} \neq \min f_{k,j}^{min}$ **then**
      do *switching* operation, $N_s \leftarrow N_s + 1$ and update $[a_{i,j}^{(ot)}]_{N\times M}$.
    end if
    $k \leftarrow k + 1$;
  until $k = \left|\mathcal{U}_F^{(p1)}\right|$
  $ot \leftarrow ot + 1$;
until $N_s \neq 0$
Output $[a_{i,j}^{(p3)}]_{N\times M} = [a_{i,j}^{(ot)}]_{N\times M}$

consumption $E_{i,j}^{C}$ is proportional to the allocated computing resource $f_{i,j}$. Thus, by substituting (11) into (9), a lower bound of the total energy consumption can be expressed as

$$E_{tot} \geq \tilde{E}_{tot} = \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{M}} a_{i,j}\left(\frac{p^{Tr}\cdot D_i}{r_{i,j}} + \kappa_j F_i (f_{i,j}^{min})^{\nu_j - 1}\right), \tag{17}$$

and the energy minimization problem can be therefore shown as

$$\begin{aligned} \mathcal{P}3: \quad & \min_{\{a_{i,j}\}} \quad \tilde{E}_{tot} \\ \text{subject to}: \ & (1), (2), (13), (14); \\ & \sum_{i\in\mathcal{N}}\sum_{j\in\mathcal{M}} a_{i,j} = |\mathcal{U}_F|, \end{aligned} \tag{18}$$

where $\mathcal{U}_F$ denotes the largest feasible task set obtained by the ITA algorithm and $|\cdot|$ denotes the cardinality of a set.

In general, with any given feasible task set $\mathcal{U}_F$, (2) and (18) can be integrated into one constraint, shown as

$$\sum_{j\in\mathcal{M}} a_{i,j} = 1, \ \forall i \in \mathcal{U}_F, \tag{19}$$

and $\mathcal{P}3$ with given feasible task set $\mathcal{U}_F$, can be seen as a typical general assignment problem, which is NP-hard. To solve $\mathcal{P}3$, a low-complexity energy minimization algorithm is developed in the following subsection.

B. Algorithm Design

Based on the feasible task allocation matrix, $[a_{i,j}^{(p1)}]_{N\times M}$, obtained via ITA algorithm, the largest feasible task set is defined as $\mathcal{U}_F^{(p1)} = \{U_f : \sum_{j\in\mathcal{M}} a_{f,j}^{(p1)} = 1, \forall U_f \in \mathcal{U}\}$. For

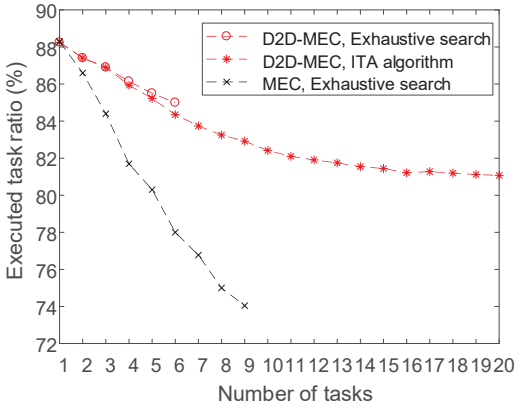

Fig. 2. Executed task ratio versus the number of tasks in traditional MEC networks and D2D-assisted MEC (D2D-MEC) networks with various admission algorithms.

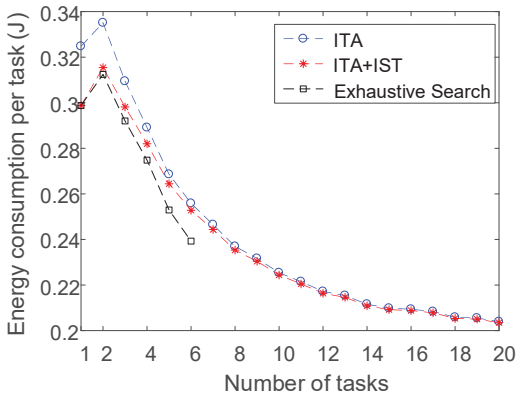

Fig. 3. Energy consumption per executed task versus the number of tasks in D2D-assisted MEC networks.

each task U_f, it can be seen from (11) that with known data rate between entity f and entity j, $r_{f,j}$, $f_{f,j}^{min}$ is a deterministic parameter, which monotonically decreases as $r_{f,j}$ increases.

Moreover, (5) shows that $E_{i,j}^{C}$ is a monotonically increasing function of $f_{i,j}^{min}$, as $(\nu_j - 1)$ is always larger than 1. Therefore, it can be easily observed that for one feasible task U_f, the energy consumption at the feasible entity j, $E_f = E_{f,j}^{Tr} + E_{f,j}^{C}$, increases with $f_{f,j}^{min}$. That is, according to (17), $\tilde{E}_{tot}$ becomes larger when the entity j with a larger $f_{f,j}^{min}$ is chosen. Note that E_f is feasible only when $f_{i,j}^{min} > 0$, as shown in (14).

Based on the above analysis, an iterative *switching* task (IST) algorithm is then proposed to minimize the energy consumption of the largest feasible task set $\mathcal{U}_F^{(p1)}$. Specifically, the *switching* operation is the switching of each task U_i from entity j, with $a_{i,j}^{(p1)} = 1$, to entity j'. There are two necessary conditions need to be satisfied to form a possible *switching* pair (j, j') for task U_i: (1) $f_{i,j'}^{min} < f_{i,j}^{min}$; and (2) entity j' must have enough computing resource left to fulfil $f_{i,j'}^{min}$. That is,

$$f_{j'}^{Re} = f_{j'} - \sum_{U_l \in \mathcal{U}_F^{(p1)}} a_{l,j'} \cdot f_{l,j'}^{min} \geq f_{i,j'}^{min}, \qquad (20)$$

where $f_{j'}^{Re}$ denotes the residual computing resource at the computing entity $j', \forall j' \in \mathcal{M}$.

For each task $U_f \in \mathcal{U}_F^{(p1)}$, if there exists multiple entities to pair with entity j, denoted by $\{j'_o : o = 1, \cdots, O\}$, that satisfy the condition of *switching* operation, task U_f is assigned to entity j^* with the minimum f_{f,j'_o}^{min}, i.e., $j^* = \arg\min f_{f,j'_o}^{min}$.

After $\left|\mathcal{U}_F^{(p1)}\right|$ iterations, the task allocation matrix $[a_{i,j}]_{N \times M}$ can be obtained. However, since only one task is checked for the *switching* operation at one iteration, if *switching* operation for task U_k is done at iteration k, the previous $k-1$ tasks might also have available switching pairs since the residual computing resource for the two switched entities is changed. Therefore, an outer iteration is needed if *switching* operation is performed during the $\left|\mathcal{U}_F^{(p1)}\right|$ iterations. The IST algorithm is summarized in Algorithm 2.

V. Simulation Results

In this section, simulation results are presented to show the benefits of D2D-assisted MEC networks and evaluate the performance of the proposed algorithms. It is assumed that N UEs are uniformly distributed in the cell with radius 100 meters and the edge AP is located at the centre of the cell. The edge cloud AP is assumed to have $f_0 = 20$GHz computing capability. Each user i has random computing capability f_i varying from 0GHz to 10GHz, for all $i \in \{1, \cdots, N\}$ and the transmit power is fixed at 10^{-3} watt. Each user randomly generates one task with $D_i = 1$, and $T_{i,max} = 1, \forall i \in \mathcal{N}$. The required CPU cycles F_i is randomly distributed between 0GHz to 10GHz. Rayleigh small scale fading is assumed with zero mean and unit variance.

A. Maximum Executed Task Ratio δ

Fig. 2 presents the executed task ratio, defined as $\delta = \frac{|\mathcal{U}_F|}{N}$, in traditional MEC networks and D2D-assisted MEC networks. The optimal maximum executed task ratio of both MEC and D2D-assisted MEC networks are obtained by using the exhaustive search method. Specifically, each task can be executed by its own UE or the edge AP for traditional MEC networks, otherwise the task cannot be accomplished. By exhaustively searching through δ of all the possible 3^N combinations for N tasks, the maximum executed task ratio of traditional MEC networks is obtained. Similarly, the maximum task executed ratio of the D2D-assisted MEC networks is obtained by exhaustively searching through δ of all the possible N^{N+2} combinations as each task have $(N+2)$ allocation options. As the exhaustive search method has computation complexity of $O(N^{N+2})$ for D2D-assisted MEC networks and $O(3^N)$ for MEC networks, the simulation results are obtained up to $N = 9$ for MEC networks and up to $N = 6$ for D2D-assisted MEC networks. It is shown in Fig. 2 that ITA algorithm can achieve nearly the same executed task ratio as the optimal exhaustive search. It is clearly shown that with the assistance of D2D communication, the executed ratio converges gently to 81% when $N = 20$. By contrast, the executed task ratio rapidly drops to 74% in traditional MEC network when $N = 9$.

It can be foreseen that in dense user scenario, although the computing resource of AP is much more than that of an

individual UE, it cannot fulfil the large quantity of offloading requests. By contrast, in D2D-assisted MEC network, the executed task ratio is much higher than that in the traditional MEC network. In fact, it is almost converges to more than 80% in dense user scenario, indicating that the user experience can be improved significantly.

B. Energy Consumption per Executed Task

Fig. 3 presents the energy consumption per executed task as a function of the number of tasks in D2D-assisted MEC networks. In general, it can be seen that the simulation results of ITA+IST decreases dramatically from 0.318 J at $N = 2$ to 0.203 J at $N = 20$. It indicates that the energy consumption per executed task reduces significantly in D2D-assisted MEC networks as the number of UEs increases. The increment of energy consumption from $N = 1$ to $N = 2$ is due to the fact that more power is desired in D2D-assisted MEC networks as more tasks are executed. However, this effect will be mitigated as the number of UEs increases. The optimal energy consumption results are obtained via exhaustive search from $N = 1$ to $N = 6$. It is shown that with the supplementary IST algorithm adopted after ITA algorithm, the energy consumption is greatly reduced and close to the optimal results when the number of tasks is small. Fig. 3 further shows that the performance of ITA algorithm converges to ITA+IST algorithms in dense user scenario. This is because when the number of tasks increases, almost full system computing resource is utilized, leading to little margin for further energy optimization.

VI. Conclusions

In this paper, the task feasibility problem and the energy minimization problem in D2D-assisted MEC network are investigated. Specifically, a low-complexity ITA algorithm is developed to maximize the number of executed tasks with the latency constraint of each individual task and the computing resource constraint of each computing entity. A supplementary IST algorithm is then proposed for global energy minimization. Simulation results show that combining ITA and IST algorithms can achieve close performance to the optimal exhaustive search method. Moreover, it is further shown that the D2D-assisted MEC network achieves higher executed task ratio than the traditional MEC network, and the energy consumption for executing each task is significantly reduced, especially in dense user scenarios.

Acknowledgment

Junyuan Wang would like to acknowledge the support of Edge Hill RIF Conference Travel Grant Award.

References

[1] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surveys Tutorials*, vol. PP, no. 99, pp. 1–1, 2017.

[2] W. Zhang, Y. Wen, K. Guan, D. Kilper, H. Luo, and D. Wu, "Energy-optimal mobile cloud computing under stochastic wireless channel," *IEEE Trans. Wireless Commun.*, vol. 12, no. 9, pp. 4569–4581, Sept. 2013.

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

[4] Y. Mao, J. Zhang, S. H. Song, and K. B. Letaief, "Stochastic joint radio and computational resource management for multi-user mobile-edge computing systems," *IEEE Trans. Wireless Commun.*, vol. 16, no. 9, pp. 5994–6009, Sept 2017.

[5] 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, June 2015.

[6] K. Habak, M. Ammar, K. A. Harras, and E. Zegura, "Femto clouds: Leveraging mobile devices to provide cloud service at the edge," in *Proc. IEEE CLOUD*, June 2015, pp. 9–16.

[7] C. You and K. Huang, "Mobile cooperative computing: Energy-efficient peer-to-peer computation offloading," *CoRR*, vol. abs/1704.04595, 2017. [Online]. Available: http://arxiv.org/abs/1704.04595

[8] G. Lee, W. Saad, and M. Bennis, "An online optimization framework for distributed fog network formation with minimal latency," *CoRR*, vol. abs/1710.05239, 2017. [Online]. Available: http://arxiv.org/abs/1710.05239

[9] X. Wang, X. Chen, W. Wu, N. An, and L. Wang, "Cooperative application execution in mobile cloud computing: A stackelberg game approach," *IEEE Communications Letters*, vol. 20, no. 5, pp. 946–949, May 2016.

[10] L. Pu, X. Chen, J. Xu, and X. Fu, "D2d fogging: An energy-efficient and incentive-aware task offloading framework via network-assisted d2d collaboration," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 12, pp. 3887–3901, Dec 2016.

[11] Y. Kai, J. Wang, H. Zhu, and J. Wang, "Resource allocation and performance analysis of cellular-assisted ofdma device-to-device communications," *IEEE Trans. Wireless Commun.*, vol. 18, no. 1, pp. 416–431, Jan 2019.

[12] J. Wang, H. Zhu, N. J. Gomes, and J. Wang, "Frequency reuse of beam allocation for multiuser massive mimo systems," *IEEE Trans. Wireless Commun.*, vol. 17, no. 4, pp. 2346–2359, April 2018.

[13] J. Wang, H. Zhu, L. Dai, N. J. Gomes, and J. Wang, "Low-complexity beam allocation for switched-beam based multiuser massive mimo systems," *IEEE Trans. Wireless Commun.*, vol. 15, no. 12, pp. 8236–8248, Dec 2016.

[14] H. Zhu, "Performance comparison between distributed antenna and microcellular systems," *IEEE J. Sel. Areas Commun.*, vol. 29, no. 6, pp. 1151–1163, June 2011.

[15] J. Wang, H. Zhu, and N. J. Gomes, "Distributed antenna systems for mobile communications in high speed trains," *IEEE J. Sel. Areas Commun.*, vol. 30, no. 4, pp. 675–683, May 2012.

[16] H. Zhu and J. Wang, "Radio resource allocation in multiuser distributed antenna systems," *IEEE J. Sel. Areas Commun.*, vol. 31, no. 10, pp. 2058–2066, October 2013.

[17] H. Zhu and J. Wang, "Performance analysis of chunk-based resource allocation in multi-cell ofdma systems," *IEEE J. Sel. Areas Commun.*, vol. 32, no. 2, pp. 367–375, February 2014.

[18] H. Zhu, "Radio resource allocation for ofdma systems in high speed environments," *IEEE J. Sel. Areas Commun.*, vol. 30, no. 4, pp. 748–759, May 2012.

[19] H. Zhu and J. Wang, "Chunk-based resource allocation in ofdma systemspart ii: Joint chunk, power and bit allocation," *IEEE Transactions on Communications*, vol. 60, no. 2, pp. 499–509, February 2012.

[20] ——, "Chunk-based resource allocation in ofdma systems - part i: chunk allocation," *IEEE Transactions on Communications*, vol. 57, no. 9, pp. 2734–2744, Sep. 2009.

[21] G. Fodor, E. Dahlman, G. Mildh, S. Parkvall, N. Reider, G. Miklos, and Z. Turnyi, "Design aspects of network assisted device-to-device communications," *IEEE Communications Magazine*, vol. 50, no. 3, pp. 170–177, March 2012.

[22] L. Yang, J. Cao, S. Tang, T. Li, and A. Chan, "A framework for partitioning and execution of data stream applications in mobile cloud computing," in *Proc. IEEE CLOUD*, June 2012, pp. 794–802.

[23] K. Wang, K. Yang, and C. Magurawalage, "Joint energy minimization and resource allocation in C-RAN with mobile cloud," *IEEE Trans. Cloud Computing*, vol. PP, no. 99, pp. 1–1, 2016.