Task Assignment Algorithms in Data Shared Mobile Edge Computing Systems

Siyao Cheng, Zhenyue Chen, Jianzhong Li, Hong Gao
School of Computer Science and Tech, Harbin Institute of Technology
Emails: csy@hit.edu.cn, chenzyhit@126.com, lijzh@hit.edu.cn, honggao@hit.edu.cn

***Abstract*—The appearance of the Mobile Edge Computing (MEC) successfully solves the bottlenecks of traditional Cloud based networks as computation ability of each mobile edge is sufficiently utilized. Since the mobile edges, including mobile devices and base stations, have certain data processing abilities, it is not necessary to offload all the computation tasks to the remote could for handling. Therefore, it is quite important to decide the optimal task assignment in a MEC system, and a series of algorithms have been proposed to solve it. However, the existing algorithms ignored the data distribution during task assignment, so that the applied ranges of these algorithms are quite limit. Considering the data sharing is very important and common in a MEC system, this paper studies the task assignment algorithm in Data Shared Mobile Edge Computing Systems, and three algorithms are proposed to deal with holistic tasks and divisible tasks, respectively. The theoretical analysis on the hardness of the problem, the correctness, complexities and ratio bounds of the algorithms are also provided. Finally, the extensive experiment results were carried out. Both of the theoretical analysis and experiment results show that our algorithms have high performance in terms of latency and energy consumption.**

I. Introduction

With the development of IoT techniques and widely expansion of mobile devices [1] [2] [3], the applications of current wireless networks become more complicated, such as the intelligent manufacture, the driverless vehicles, smart health care.*etc*. Meanwhile, the QoS(Quality of Service) requirements desired by users, including the response delay, the computation accuracy, the congestion control, and the energy conservation *.etc*, become much stricter than before. Obviously, such requirements cannot be satisfied well in traditional Cloud based networks, where all the data need to be transmitted to the Cloud for processing. Furthermore, the data sampled by current IoT devices manifest an explosive growth, so that it is also not practical to transmit all the data to the Cloud.

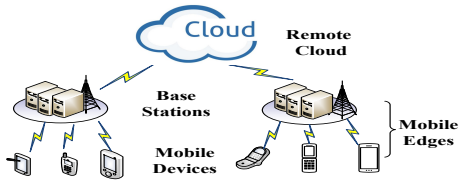


Fig. 1: The three levels of a MEC system

Fortunately, the appearance of the Mobile Edge Computing (MEC) provides a feasible way to solve the above bottlenecks. In a MEC system, the computation abilities of each mobile device and based station are sufficiently utilized, and thus it dramatically decreases the response delay as well as the transmission burden of the network. Similar to many previous studies [4] [5], the architecture of a MEC system considered in our paper have three levels as shown in Fig.1. The first level consists of a number of mobile devices, such as mobile phones, smart sensors *.etc*. These devices are mainly responsible for sampling data and communicating with users. The second level of a MEC system contains a series of base stations, in which a small scale could can be deployed. The base station and the surrounding mobile devices are connected by radio access networks, and they forms the mobile edges for a MEC system. The last level is usually to be the remote cloud which own the richer resources and have higher data processing abilities comparing with the mobile edges, which includes the mobile devices in first level and base stations in second level. However, the response latency is quite long and the energy cost is very high if all computation task are offloaded to the cloud for processing.

Since each mobile edge has certain data processing abilities, it is also not necessary to offload all the tasks to the remote could for handling. Based on such motivation, a series of task assignment algorithms have been proposed in MEC systems. The early ones investigate the problem of offloading the computation tasks for a single user in MEC systems. To solve it, [6] and [7] proposed the Binary offloading algorithm to optimize the latency and energy cost. These works firstly utilized the processing abilities of the mobile edges to improve the performance of the whole network. However, the situation discussed by them seems a little simple. Considering that a MEC system always serves multiple users, [8] has formulated the computation offloading problem of multiple users via a single base station as a computation offloading game, and design a distributed computation offloading mechanism that can achieve the Nash equilibrium to solve it. In [9], the authors have explored a distributed task offloading algorithm for multiple users in multi-channel wireless networks. The above two methods are more practical comparing with the early ones, however, the cooperations among different base stations are not considered. Due to such reason, a new framework that allow a mobile device to offload the tasks to multiple base stations was studied by [10], and semidefinite relaxation-based algorithms were provided to determine the task assignment. However, the situation of serving multiple users has not been addressed sufficiently. More Recently, a series of new task offloading algorithms that serve for multiple

users with multiple tasks in a MEC system containing three levels were studied by [11], [12] and [13]. Such algorithms make full use of the MEC systems, however, rare of them take the data distribution and task deadline constraint into account during the task assignment.

As the necessary input, the data required by a computation task may be distributed in different mobile devices or even in different clusters. For example, in a intelligent traffic monitoring system, a user want to know the average flow rate of vehicles in the whole city, while the data sampled by his mobile device only show the vehicle flow rate in a small region. In a object tracking system, a mobile device is required to return the whole trajectory of the monitoring object, while it only has partial trajectory information. Therefore, the data sharing is quite important and common in a MEC system, and thus, the task assignment problem in such Data Shared Mobile Edge Computing System needs to be studied sufficiently. Furthermore, the limited computation abilities of mobile devices and base stations, the deadline constraint of each task are also ignored by the existing methods, which are very important during task assignment in a MEC system as well.

To overcome the above problems, this paper studied the task assignment algorithm in a Data-Shared MEC system. Considering the properties of the tasks, we distinguish the tasks into two categories, which are the holistic tasks and the divisible tasks. For the holistic tasks, the raw data transmission is inevitable since they cannot be processed distributedly, thus, we try to minimize the whole energy cost by transmission and computation. For the divisible tasks, which can be processed in distributed manner, we try to avoid the raw data transmission by migrating the computation and rearranging the tasks. Since the computation operations and partial results are much smaller than the raw data, lots of energy will be saved. In summary, the main contributions are as follows.

(1) The task assignment problem is firstly considered and formally defined in Data-Shared MEC Systems.

(2) For the holistic tasks, it is proved to be NP-complete for deciding the optimal task assignment in MEC systems. An approximation algorithm based on linear programming is proposed to solve it, and the detailed analysis on the correctness, the complexity and the ratio bound, is also given.

(3) For the divisible tasks, two algorithms are provided to meet different optimization goals, and the performance of the algorithms are also analyzed theoretically.

(4) The extensive experiments were carried out, where the experimental results show that all proposed algorithms were efficient in terms of latency and energy consumption.

The organization is as follows. Section II provides the system model and problem definitions. Sections III and IV gives task Assignment Algorithms for Holistic and Divisible tasks. Section V shows the experiment results. Section VI surveys related works and Section VII concludes the paper.

II. System Model

Without loss of generality, we assume that there exist k base stations,denoted by $B_1, B_2, ..., B_k$, and n users which are $U_1, U_2, ..., U_n$, in a MEC system. Each user U_i $(1 \leq i \leq n)$ has a corresponding mobile device, denoted by i. Each base station B_r $(1 \leq r \leq k)$ connects n_r mobile devices by radio access network, and these mobile devices form a cluster, where $\sum_{r=1}^{k} n_r = n$. Similar to the existing works such as [9], we also consider the quasi-static scenario during the task assignment, *i.e.* each mobile device connects the same base station in the period we considered.

According to the discussion in Section I, each mobile device will collect the computation tasks form users. For a computation task $\mathcal{T}_{ij}$(such as some data partitioned oriented task), which is the j-th computation task raised by user U_i, we use op_{ij} to denote the operations of task $\mathcal{T}_{ij}$ $(1 \leq i \leq n, 1 \leq j \leq m)$. Generally, a mobile device may not own all the data required by its computation tasks. Therefore, the input data of $\mathcal{T}_{ij}$ can be divided into two separated subsets, the local data LD_{ij} and the estimated external data ED_{ij}. Let $\alpha_{ij} = |LD_{ij}|$, $\beta_{ij} = |ED_{ij}|$, and L_{ij} denote the possible cluster (or mobile devices) that may own ED_{ij}. Besides the input data, each computation task also occupies some resources (*e.g.* memory) and has the deadline to meet, which are denoted by C_{ij} and T_{ij}, respectively. Thus, a computation task $\mathcal{T}_{ij}$ can be regarded as a tuple, that is, $\mathcal{T}_{ij} = (op_{ij}, LD_{ij}, ED_{ij}, L_{ij}, C_{ij}, T_{ij})$. For the convenience of discussion, we assume that every user raises the same number of tasks to a MEC system. All proposed algoritnms are also suitable for dealing with the opposite situation with minor revisions.

When the tasks are holistic, then all the local and external data should be collected to a single subsystem for processing, that is a computation task $\mathcal{T}_{ij}$ $(1 \leq i \leq n, 1 \leq j \leq m)$ could be carried out by one of three possible subsystems, *i.e.* the mobile device i, the base station connected to i or the remote cloud. We use the indicator variable x_{ijl} to show which subsystem is involved to deal with $\mathcal{T}_{ij}$, and x_{ijl} satisfies that

$$x_{ijl} = \begin{cases} 1, & \text{If } \mathcal{T}_{ij} \text{ is processed by subsystem } l \\ 0, & \text{Otherwise} \end{cases} \quad (1)$$

where Subsystem 1 (*i.e.* $l = 1$) means the mobile device i, Subsystem 2 (*i.e.* $l = 2$) is the base station connected to i and Subsystem 3 (*i.e.* $l = 3$) is regarded as the remote cloud.

Basically, the objective of assigning the computation tasks in a MEC system is to determine proper $\{x_{ijl} | 1 \leq i \leq n, 1 \leq j \leq m, l \in \{1, 2, 3\}\}$ which can minimize the energy cost of the system on condition that the deadlines of the tasks could be satisfied. There exist two main factors, *i.e.* computation and transmission, that influences the runtime and energy cost of each task. Therefore, we will show the models of computation and transmission in the following two subsections.

A. Computation Model

Since the first situation we considered is that the computation tasks are indivisible and holistic, only one subsystem is chosen to carried out $\mathcal{T}_{ij}$, so that $x_{ij1} + x_{ij2} + x_{ij3} = 1$. The situation that computation tasks are divisible will be considered in Section IV.

Let f_i represent the CPU frequency of mobile device i, $\lambda_{ij1}(y)$ be the function to define CPU cycles to process $\mathcal{T}_{ij}$ on condition that the input data size is y, where $\lambda_{ij1}(y)$ can be determined by the computation complexity of $\mathcal{T}_{ij}$. Thus, if $x_{ij1} = 1$, *i.e.* task $\mathcal{T}_{ij}$ is processed locally, then the computation time and energy cost for processing $\mathcal{T}_{ij}$ can be estimated by the following formulas,respectively.

$$t_{ij1}^{(C)} = \frac{\lambda_{ij1}(\alpha_{ij}+\beta_{ij})}{f_i},\ E_{ij1}^{(C)} = \kappa\lambda_{ij1}(\alpha_{ij}+\beta_{ij})f_i^2 \quad (2)$$

according to [6] and [14], where κ is a constant related to the hardware architecture.

Similarly, if $x_{ij2} = 1$ or $x_{ij3} = 1$, that is, the base station which connects with mobile device i or the remote cloud is selected to deal with task $\mathcal{T}_{ij}$, the computation time $t_{ij2}^{(C)}$ and $t_{ij3}^{(C)}$ can be determined by the following formulas,respectively.

$$t_{ij2}^{(C)} = \frac{\lambda_{ij2}(\alpha_{ij}+\beta_{ij})}{f_s},\ t_{ij3}^{(C)} = \frac{\lambda_{ij3}(\alpha_{ij}+\beta_{ij})}{f_c} \quad (3)$$

where $\lambda_{ij2}(\alpha_{ij}+\beta_{ij})$ and $\lambda_{ij3}(\alpha_{ij}+\beta_{ij})$ represents the CPU cycles of a base station and the cloud for dealing with the data with $\alpha_{ij}+\beta_{ij}$ size, f_s and f_c are the CPU frequencies of a base station and the cloud, respectively.

Since the energy cost of base station and the cloud during computation is extremely small comparing with that cost by transmission, therefore, they can be ignored.

B. Transmission Model

In order to obtain all the data for a task, each mobile device should exchange the data information with the connected base station by radio access network.

Let $r_i^{(U)}$ and $r_i^{(D)}$ be the upload and download rate of mobile device i $(1 \le i \le n)$. Both of them can be determined by [9] [10] using the Shannon Theory, that is $r_i^{(U)} = W_i^{(U)}\log_2(1+\frac{g_i^{(U)}P_i^{(T)}}{\varpi_0}), r_i^{(D)} = W_i^{(D)}\log_2(1+\frac{g_i^{(D)}P^{(S)}}{\varpi_0})$ where $g_i^{(U)}$ and $g_i^{(D)}$ are the uplink and downlink channel gains between mobile device i and the base station; $W_i^{(U)}$ and $W_i^{(D)}$ are the uplink and downlink channel bandwidths that the base station allocates for the mobile device i; $P_i^{(T)}$ and $P^{(S)}$ are the transmission powers of mobile device i and the base station; ϖ_0 is the power of the white noise.

Let $e_i^{(T)}(X)$, $e_i^{(R)}(X)$ and $e_{B,B}(X)$ denote energy cost of transmitting data with size X from mobile device i to the base station, from the base station to mobile device i, between two base stations, respectively. All of them also can be determined by the transmission rate and power according to [9].

Thus, if $x_{ij1} = 1$, then the mobile device i should retrieve the external data with the size β_{ij} from other mobile devices through the MEC system. Based on the definition of the task, L_{ij} will indicate which mobile device may own such external data. Therefore, the data retrieving time $t_{ij1}^{(R)}$ can be determined by the following formula.

$$t_{ij1}^{(R)} = \begin{cases} \frac{\beta_{ij}}{r_{L_{ij}}^{(U)}} + \frac{\beta_{ij}}{r_i^{(D)}}, & \text{If } L_{ij},\ i \text{ are in the same cluster} \\ \frac{\beta_{ij}}{r_{L_{ij}}^{(U)}} + \frac{\beta_{ij}}{r_i^{(D)}} + t_{B,B}(\beta_{ij}), & \text{Otherwise} \end{cases}$$

where $t_{B,B}(\beta_{ij})$ is the time spent for transmitting the data with size β_{ij} between two base stations.

The energy cost of retrieving the external data, $E_{ij1}^{(R)}$, satisfies that

$$E_{ij1}^{(R)} = \begin{cases} e_{L_i}^{(T)}(\beta_i) + e_i^{(R)}(\beta_i), & \text{If } L_{ij},\ i \text{ are in same cluster} \\ e_{L_i}^{(T)}(\beta_i) + e_i^{(R)}(\beta_i) + e_{B,B}(\beta_i) & \text{Otherwise} \end{cases}$$

Secondly, if $x_{ij2} = 1$, both of the local data and external data of $\mathcal{T}_{ij}$ should be transmitted to the base station for processing, and the result should be returned to mobile device i. Suppose that $\eta(y)$ represent the size of the computation result on condition that the input data size equals to y, then the information transmission time, $t_{ij2}^{(R)}$ satisfies the following formula.

$$t_{ij2}^{(R)} = \begin{cases} \max\{\frac{\beta_{ij}}{r_{L_{ij}}^{(U)}}, \frac{\alpha_{ij}}{r_i^{(U)}}\} + \frac{\eta(\alpha_{ij}+\beta_{ij})}{r_i^D}, \\ \qquad \text{If } L_{ij} \text{ and } i \text{ are in the same cluster} \\ \max\{\frac{\beta_{ij}}{r_{L_{ij}}^{(U)}} + t_{B,B}(\beta_j), \frac{\alpha_{ij}}{r_i^{(U)}}\} + \frac{\eta(\alpha_{ij}+\beta_{ij})}{r_i^D}, \text{ Else} \end{cases}$$

The total energy cost during transmission, denoted by $E_{ij2}^{(R)}$, can be calculated by Formula (4).

$$E_{ij2}^{(R)} = \begin{cases} e_{L_i}^{(T)}(\beta_i) + e_i^{(T)}(\alpha_i) + e_i^{(R)}(\eta(\alpha_{ij}+\beta_{ij})), \\ \qquad \text{If } L_{ij},\ i \text{ are in same cluster} \\ e_{L_i}^{(T)}(\beta_i) + e_i^{(T)}(\alpha_i) + e_i^{(R)}(\eta(\alpha_{ij}+\beta_{ij})) + e_{B,B}(\beta_i) \\ \qquad \text{Otherwise} \end{cases} \quad (4)$$

Thirdly, if $x_{ij3} = 1$, that is, the remote cloud is selected to deal with $\mathcal{T}_{ij}$. Thus, both of the local data and the external data are required to be delivered to the cloud for further processing. Let $t_{B,C}(X)$ and $e_{B,C}(X)$ be the transmission delay and energy cost of transmitting data with size X.

Then, the time and energy spent during transmission, denoted by $t_{ij3}^{(R)}$ and $E_{ij3}^{(R)}$, can be determined by the two formulas when $x_{ij3} = 1$.

$$t_{ij3}^{(R)} = \max\{\frac{\beta_{ij}}{r_{L_{ij}}^{(U)}}, \frac{\alpha_{ij}}{r_i^{(U)}}\} + \frac{\eta(\alpha_{ij}+\beta_{ij})}{r_i^D} + t_{B,C}(\alpha_{ij}+\beta_{ij}+\eta(\alpha_{ij}+\beta_{ij}))$$

$$E_{ij3}^{(R)} = e_{L_i}^{(T)}(\beta_i) + e_i^{(T)}(\alpha_i) + e_i^{(R)}(\eta(\alpha_{ij}+\beta_{ij})) + e_{B,C}(\alpha_{ij}+\beta_{ij}+\eta(\alpha_{ij}+\beta_{ij}))$$

Based on the above analysis, more data are required to transmit when $x_{ij3} = 1$, and the transmission time from a base station to the remote cloud is much larger than that between two base stations [15] [16], so that more energy will be consumed when $x_{ij3} = 1$, that is $E_{ij3}^{(R)} > E_{ij2}^{(R)}$.

C. Problem Definition

Based on the computation and transmission models, the total delay and energy cost for processing task $\mathcal{T}_{ij}$ when $l = 1, 2, 3$ are given as follows.

$$t_{ijl} = t_{ijl}^{(C)} + t_{ijl}^{(R)}, E_{ijl} = \begin{cases} E_{ijl}^{(R)} + E_{ijl}^{(C)} & \text{if } l = 1 \\ E_{ijl}^{(R)} & \text{if } l = 2, 3 \end{cases} \quad (5)$$

Meanwhile, the resources (*e.g.* memory, thread, virtual machine, processor) provided by mobile devices and base stations for computation are limited as discussed in Section I, so that we use max_i and max_S to denote upper bound of such resource. Thus, $\sum_{j=1}^{m} C_{ij}x_{ij1} \leq max_i$ and $\sum_{i=1}^{n}\sum_{j=1}^{m} C_{ij}x_{ij2} \leq max_S$, which means that the total resource occupied by the tasks assigned to mobile device i and a base station should be no more than max_i and max_S.

Thus, the problem of Holistic Task Assignment (HTA) in a MEC system is defined as follows.

Input:

1) A MEC system with n mobile devices,$\{1, 2, ...n\}$, and k base stations, $\{B_1, B_2, ..., B_k\}$;
2) A set of computation tasks $\{\mathcal{T}_{ij}|1 \leq i \leq n, 1 \leq j \leq m\}$, and the estimated CPU cycles and result size for each computation task.
3) Basic frequencies of mobile devices, base station and the cloud, $\{f_i|1 \leq i \leq n\}$, f_s, f_c;
4) The Network parameters, $r_i^{(U)}$, $r_i^{(D)}$, $t_{B,B}(X)$, $t_{B,C}(X)$ $e_i^{(T)}(X)$, $e_i^{(R)}(X)$, $e_{B,B}(X)$ and $e_{B,C}(X)$;
5) The limitations on computation resources of each mobile devices and the base station $\{max_i|1 \leq i \leq n\}$, max_S.

Output: The task assignment strategy $\{x_{ijl}|1 \leq i \leq n, 1 \leq j \leq m, l = 1, 2, 3\}$, where the total energy cost of the system, *i.e.* $\sum_{i=1}^{n}\sum_{j=1}^{m}\sum_{l=1}^{3} E_{ijl}x_{ijl}$, is minimized, and the following conditions are satisfied.

$$\sum_{l=1}^{3} t_{ijl}x_{ijl} \leq T_{ij} \quad \forall i \in N, \forall j \in M, \tag{C1}$$

$$\sum_{j=1}^{m} C_{ij}x_{ij1} \leq max_i \quad \forall i \in N, \tag{C2}$$

$$\sum_{i=1}^{n}\sum_{j=1}^{m} C_{ij}x_{ij2} \leq max_S \tag{C3}$$

$$\sum_{l=1}^{3} x_{ijl} = 1 \quad \forall i \in N, \forall j \in M, \tag{C4}$$

$$x_{ijl} \in \{0, 1\}, \forall i \in N, \forall j \in M, \forall l \in \{1, 2, 3\} \tag{C5}$$

The hardness and solution of such problem will be addressed in Section III.

III. Holistic Task Assignment Algorithm

Before introducing the algorithm, we firstly analyze the hardness of HTA problem.

Theorem 1. *The HTA problem is NP-complete.*

Proof Considering a special case of the HTA problem, that is, $max_i = 0$, $T_{ij} = \infty$ for all $1 \leq i \leq n$ and $1 \leq j \leq m$, which the deadline of each task can be ignored and the mobile devices does not have any ability for processing tasks.

Therefore, we have that $x_{ij1} = 0$ and $x_{ij3} = 1 - x_{ij2}$ and the HTA problem is equal to the following one

$$\begin{aligned} P1: \max \ & \sum_{i=1}^{n}\sum_{j=1}^{m}(E_{ij3} - E_{ij2})x_{ij2} \\ s.t. \ & \sum_{i=1}^{n}\sum_{j=1}^{m} C_{ij}x_{ij2} \leq max_S \\ & x_{ij2} \in \{0, 1\}, \forall i \in N, \forall j \in m \end{aligned}$$

Since $E_{ij3} > E_{ij2}$, the problem $P1$ can be regarded as a Knapsack problem with $n \times m$ items, the value and weight of item (i, j) equals to $E_{ij3} - E_{ij2}$ and C_{ij}, and the capacity of Knapsack is max_S. Since Knapsack problem is NP-complete, that is, the special case of the HTA problem is NP-complete, so that HTA problem is at least NP-complete. □

Since HTA problem is NP-complete, the following subsection will proposed an approximation algorithm to solve it.

A. Linear Programming based Algorithm

Based on Section II, there only exist three possible subsystems to process task $\mathcal{T}_{ij}$, which are mobile device i, the connected base station B_r and the remote cloud. Therefore, each cluster can be considered separately, so that this section will discuss how to assignment the task in a cluster.

Let $(\mathbf{y_1}, \mathbf{y_2}, ..., \mathbf{y_k})$ denoted the combination of k vectors. For example, $(\mathbf{y}, \mathbf{z}) = (y_1, y_2, ..., y_r, z_1, z_2, ..., z_l)$ if $\mathbf{y} = (y_1, y_2, ..y_r)$ and $\mathbf{z} = (z_1, z_2., , , z_l)$. Thus, we use $\xi_{ij} = (x_{ij1}, x_{ij2}, x_{ij3})$ to represent the assignment of task $\mathcal{T}_{ij}$, and then $\xi_i = (\xi_{i1}, \xi_{i2}, ..., \xi_{im})$ denotes the assignment result of all the tasks given by user U_i, and vector $\xi = (\xi_1, \xi_2, ..., \xi_{n_r})$ denote the assignment result of all the tasks in the cluster. By the same way, $\mathbf{e}_{ij} = (E_{ij1}, E_{ij2}, E_{ij3})$ denotes the possible energy cost of task $\mathcal{T}_{ij}$, and we set $\mathbf{e}_i = (\mathbf{e}_{i1}, \mathbf{e}_{i2}, \ldots, \mathbf{e}_{im})$, and $\mathbf{e} = (\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_{n_r})$.

Obviously, both of ξ and $\mathbf{e}$ are the vectors with $3n_rm$ size. Based on ξ and $\mathbf{e}$ and the definition of the HTA problem, the relax linear programming problem is formalized as follows.

$$\begin{aligned} P2: \min_{\xi} \ & \mathbf{e}\xi^T \\ s.t. \ & \mathbf{A}_1\xi^T \leq \mathbf{b}_1, \quad \mathbf{A}_2\xi^T \leq \mathbf{b}_2, \\ & \mathbf{A}_3\xi^T \leq \mathbf{b}_3, \quad \mathbf{A}_4\xi^T = \mathbf{b}_4, \\ & \xi[i] \in [0, 1], \forall i = 1, 2, \ldots, 3 \times n_r \times m \end{aligned}$$

where $\xi[i]$ is the i-th element of vector ξ, $\mathbf{b}_1 = (T_{11}, T_{11}, T_{11}, \ldots, T_{ij}, T_{ij}, T_{ij}, \ldots, T_{n_rm}, T_{n_rm}, T_{n_rm})^T_{1\times 3n_rm}$, $\mathbf{b}_2 = (max_1, max_2 \ldots, max_i, \ldots, max_{n_r})^T_{1\times n_r}$, $\mathbf{b}_3 = (max_S)$, $\mathbf{b}_4 = (1, 1, \ldots, 1)^T_{n_rm\times 1}$, $\mathbf{A}_1 = diag(t_{111}, t_{112}, t_{113}, ..., t_{n_rm1}, t_{n_rm2}, t_{n_rm3})_{3n_rm\times 3n_rm}$, $\mathbf{A}_3 = [0, C_{i1}, 0, 0, C_{i2}, 0, \ldots, 0, C_{n_rm}, 0]_{1\times 3n_rm}$,

$$\mathbf{A}_2 = \begin{bmatrix} \mathbf{a}_1 & \mathbf{0}_{1\times 3m} & \cdots & \mathbf{0}_{1\times 3m} \\ \mathbf{0}_{1\times 3m} & \mathbf{a}_2 & \cdots & \mathbf{0}_{1\times 3m} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0}_{1\times 3m} & \mathbf{0}_{1\times 3m} & \cdots & \mathbf{a}_{n_r} \end{bmatrix}_{n_r\times 3n_rm},$$

$$\mathbf{A}_4 = \begin{bmatrix} \mathbf{a}_{11} & \mathbf{0}_{1\times 3} & \cdots & \mathbf{0}_{1\times 3} \\ \mathbf{0}_{1\times 3} & \mathbf{a}_{12} & \cdots & \mathbf{0}_{1\times 3} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{0}_{1\times 3} & \mathbf{0}_{1\times 3} & \cdots & \mathbf{a}_{n_rm} \end{bmatrix}_{n_rm\times 3n_rm},$$

and $\mathbf{a}_i = (C_{i1}, 0, 0, \ldots, C_{im}, 0, 0)_{1\times 3m}$,and $\mathbf{a}_{ij} = (1, 1, 1)$.

Based on above symbols, the Holistic Task Assignment Algorithm for (*LP-HTA* for short) has six steps.

Step 1. Solve the linear programming problem $P2$ to obtain ξ by using the interior points method algorithm given in [17].

Step 2. Construct an equivalent fractional matrix $\mathbf{X}$ according to ξ. That is, $\mathbf{X}[i, j, l] = \xi[3m \times (i-1) + 3 \times (j-1) + l]$ for any $1 \leq i \leq n_r$, $1 \leq j \leq m$ and $l \in \{1, 2, 3\}$.

Step 3. If all elements of $\mathbf{X}$ are binary, then set $x_{ijl} = X[i,j,l]$, and return $\{x_{ijl}|1 \le i \le n_r, 1 \le j \le m, l \in \{1,2,3\}\}$ as the assignment result. Otherwise, let $q = \text{argmax}_{l\in\{1,2,3\}} \mathbf{X}[i,j,l]$, (*i.e.* $\mathbf{X}[i,j,q] = \max\{\mathbf{X}[i,j,1], \mathbf{X}[i,j,2], \mathbf{X}[i,j,3]\}$), then set

$$\widehat{x}_{ijl} = \begin{cases} 1 & \text{if } l = q \\ 0 & \text{Otherwise} \end{cases} \quad (6)$$

Step 4. Considering each $\widehat{x}_{ijq} = 1$, if $t_{ijq} \le T_{ij}$, set $x_{ijq} = 1$ and $x_{ijp} = 0$ for any $p \in \{1,2,3\}$ and $p \ne q$. Otherwise, find l which satisfies that $l = \text{argmax}_{p\in\{p|t_{ijp}\le T_{ij}\}} \mathbf{X}[i,j,p]$, then set $x_{ijl} = 1$ and $x_{ijr} = 0$ for any $r \in \{1,2,3\}$ and $r \ne l$. If we cannot find such l, then we cancel $\mathcal{T}_{ij}$ and inform users.

Step 5. For each mobile device i, reconsidering Constraint $C2$ in HTA problem. Compute $\sum_{j=1}^{m} C_{ij}x_{ij1}$. When $\sum_{j=1}^{m} C_{ij}x_{ij1} > max_i$, the following steps are carried out.

1) Select tasks from $S_1 = \{\mathcal{T}_{ij}|x_{ij1} = 1 \wedge t_{ij2} \le T_{ij}\}$ greedily according to the resource they occupy (*i.e.* C_{ij});
2) Set $x_{ij1} = 0$ and $x_{ij2} = 1$ for any selected task $\mathcal{T}_{ij}$, that is, remove the task $\mathcal{T}_{ij}$ to the base station for processing;
3) Repeat the above two steps until $\sum_{j=1}^{m} C_{ij}x_{ij1} \le max_i$ or $\{\mathcal{T}_{ij}|x_{ij1} = 1 \wedge t_{ij2} \le T_{ij}\} = \emptyset$;

If $\sum_{j=1}^{m} C_{ij}x_{ij1}$ is still larger than max_i, greedily select tasks with high resource occupation to cancel and inform users.

Step 6. For the base station, we do the similar operations as Step 5. First, check whether $\sum_{i=1}^{n_r}\sum_{j=1}^{m} C_{ij}x_{ij2} \le max_S$ is satisfied. If it is, then return the assignment result $\{x_{ijl}|1 \le i \le n_r, 1 \le j \le m, l \in \{1,2,3\}\}$. Otherwise,

1) Calculate $S_2 = \{\mathcal{T}_{ij}|x_{ij2} = 1 \wedge t_{ij3} \le T_{ij}\}$. If $S_2 \ne \emptyset$, select $\mathcal{T}_{ij}$ from S_2 greedily according to C_{ij}. Set $x_{ij2} = 0$ and $x_{ij3} = 1$, *i.e.* remove $\mathcal{T}_{ij}$ to Cloud to process.
2) Repeat the above step until $\sum_{i=1}^{n_r}\sum_{j=1}^{m} C_{ij}x_{ij2} \le max_S$ or $\{\mathcal{T}_{ij}|x_{ij2} = 1 \wedge t_{ij3} \le T_{ij}\} = \emptyset$

If $\sum_{i=1}^{n_r}\sum_{j=1}^{m} C_{ij}x_{ij2} \le max_S$ still can not be satisfied, then greedily select some tasks with high resource occupation, cancel them and inform the users.

B. Performance Analysis of the Algorithm

1) Correctness: Firstly, we need to verifies that *LP-HTA* algorithm provided a feasible solution for the HTA problem.

Suppose that $\{x_{ijl}|1 \le i \le n_r, 1 \le j \le m, l \in \{1,2,3\}\}$ be the assignment result returned by *LP-HTA* algorithm. As shown in the algorithm, we only set one of x_{ij1}, x_{ij2} and x_{ij3} to be 1, and set the other two to be 0, so that Constraints $C4$ and $C5$ in HTA problem are certainly satisfied.

Meanwhile, **Step 4** of the algorithm considers the deadline constraint of each task. That is, x_{ijl} is set to be 1 only if $t_{ijl} \le T_{ij}$, where $1 \le i \le n_r$, $1 \le j \le m$ and $l \in \{1,2,3\}$. Furthermore, the deadline constraint is also considered in **Step 5** and **Step 6** when we remove the tasks to base station or to the cloud. Thus, Constraint $C1$ of HTA problem is satisfied.

Finally, **Step 5** of the algorithm guarantees that $\sum_{j=1}^{m} C_{ij}x_{ij1} \le max_i$ for each mobile i $(1 \le i \le n_r)$, so that Constraint $C2$ is satisfied. Similarly, Constrain $C3$ in HTA problem is also meet by applying **Step 6** of algorithms

In summary, $\{x_{ijl}|1 \le i \le n_r, 1 \le j \le m, l \in \{1,2,3\}\}$ satisfies all the constraints, so it is a feasible solution for the HTA problem.

2) Complexity: Next, we will analyze the complexity of *LP-HTA* algorithm.

In **Step 1**, the interior points method is used for solve the linear programming problem and obtain ξ, which is a vector with $3n_rm$ size. The complexity is $O((n_rm)^{3.5})$ according to [17]. In **Step 2** and **Step 3**, we construct the equivalent fractional matrix $\mathbf{X}$ and $\{\widehat{x}_{ijl}|1 \le i \le n_r, 1 \le j \le m, l \in \{1,2,3\}\}$ according to ξ. The complexity of such construction is $O(n_rm)$. In **Step 4**, **Step 5** and **Step 6**, we adjust $\{\widehat{x}_{ijl}|1 \le i \le n_r, 1 \le j \le m, l \in \{1,2,3\}\}$ according the deadline and processing abilities constraints, and get $\{x_{ijl}|1 \le i \le n_r, 1 \le j \le m, l \in \{1,2,3\}\}$ finally. The complexity of such adjustment is still $O(n_rm)$. **In summary**, the total complexity of *LP-HTA* algorithm is $O((n_rm)^{3.5})$.

3) Ratio Bound: Before discussing the ratio bound of *LP-HTA* algorithm, the following lemma is proved.

Lemma 1. *The intermediate result* $\{\widehat{x}_{ijl}|1 \le i \le n_r, 1 \le j \le m, l \in \{1,2,3\}$ *calculated by Step 3 in the LP-HTA algorithm satisfies that* $\sum_{i=1}^{n_r}\sum_{j=1}^{m}\sum_{l=1}^{3} E_{ijl}\widehat{x}_{ijl} \le 3\sum_{i=1}^{n_r}\sum_{j=1}^{m}\sum_{l=1}^{3} E_{ijl}x_{ijl}^{OPT}$, *where* $\{x_{ijl}^{OPT}|1 \le i \le n_r, 1 \le j \le m, l \in \{1,2,3\}\}$ *be the optimal assignment for HTA problem.*

Proof. According to the definition of relax linear programming problem, *i.e.* Problem $P2$, we have $\mathbf{X}[i,j,1]+\mathbf{X}[i,j,2]+\mathbf{X}[i,j,3] = 1$. Therefore, $\max_{l=1,2,3} \mathbf{X}[i,j,l] \ge \frac{1}{3}$.

Let r satisfies that $\mathbf{X}[i,j,r] = \max_{l=1,2,3} \mathbf{X}[i,j,l]$. According to **Step 3**, we set $\widehat{x}_{ijr} = 1$, and $\widehat{x}_{ijl} = 0$ if $l \ne r$. Since $\mathbf{X}[i,j,r] = \max_{l=1,2,3} \mathbf{X}[i,j,l] \ge \frac{1}{3} = \frac{1}{3}\widehat{x}_{ijr}$, we have $\sum_{i=1}^{n_r}\sum_{j=1}^{m}\sum_{l=1}^{3} E_{ijl}\widehat{x}_{ijl} = \sum_{i=1}^{n_r}\sum_{j=1}^{m} E_{ijr}\widehat{x}_{ijr} \le 3\sum_{i=1}^{n_r}\sum_{j=1}^{m} E_{ijr}\mathbf{X}[i,j,r] \le 3\sum_{i=1}^{n_r}\sum_{j=1}^{m}\sum_{l=1}^{3} E_{ijl}\mathbf{X}[i,j,l]$

where $\sum_{i=1}^{n_r}\sum_{j=1}^{m}\sum_{l=1}^{3} E_{ijl}\mathbf{X}[i,j,l]$ is the optimal result of the linear programming problem.

Since $\{x_{ijl}^{OPT}|1 \le i \le n_r, 1 \le j \le m, l \in \{1,2,3\}\}$ be the optimal assignment of HTA problem, so that it is just a feasible solution of the linear programming problem. Therefore, $\sum_{i=1}^{n_r}\sum_{j=1}^{m}\sum_{l=1}^{3} E_{ijl}\mathbf{X}[i,j,l] \le \sum_{i=1}^{n_r}\sum_{j=1}^{m}\sum_{l=1}^{3} E_{ijl}x_{ijl}^{OPT}$. Thus, we have $\sum_{i=1}^{n_r}\sum_{j=1}^{m}\sum_{l=1}^{3} E_{ijl}\widehat{x}_{ijl} \le 3\sum_{i=1}^{n_r}\sum_{j=1}^{m}\sum_{l=1}^{3} E_{ijl}x_{ijl}^{OPT}$. □

According to *LP-HTA* algorithm, **Step 4**, **Step 5** and **Step 6** are used to adjust $\{\widehat{x}_{ijl}|1 \le i \le n_r, 1 \le j \le m, l \in \{1,2,3\}$ to satisfy Constraints $C1$, $C2$ and $C3$. In another words, some tasks are migrated among mobile devices, the base station and the cloud to meet the constraints. Let Δ be the growth of energy cost caused by the above migration, and $E_{LP}^{(OPT)} = \sum_{i=1}^{n_r}\sum_{j=1}^{m}\sum_{l=1}^{3} E_{ijl}\mathbf{X}[i,j,l]$ be the optimal result of the linear programming algorithm $P2$, then the ratio bound of *LP-HTA* algorithm meets the following theorem.

Theorem 2. *The ratio bound of LP-HTA Algorithm, i.e. R, is*

no more than $3+\frac{\Delta}{E_{LP}^{(OPT)}}$.

Proof. According to Lemma 1, we have $\sum_{i=1}^{n_r}\sum_{j=1}^{m}\sum_{l=1}^{3}E_{ijl}\widehat{x}_{ijl} \leq 3\sum_{i=1}^{n_r}\sum_{j=1}^{m}\sum_{l=1}^{3}E_{ijl}x_{ijl}^{OPT}$, where $\{\widehat{x}_{ijl}|1 \leq i \leq n_r, 1 \leq j \leq m, l \in \{1,2,3\}\}$ is intermediate result obtained by **Step 3** of *LP-HTA* algorithm.

Since Δ is the growth of energy cost caused by the task migration and $E_{LP}^{(OPT)} = \sum_{i=1}^{n_r}\sum_{j=1}^{m}\sum_{l=1}^{3}E_{ijl}\mathbf{X}[i,j,l] \leq \sum_{i=1}^{n_r}\sum_{j=1}^{m}\sum_{l=1}^{3}E_{ijl}x_{ijl}^{OPT}$ based on **Lemma 1**, we have

$$R = \frac{\sum_{i=1}^{n_r}\sum_{j=1}^{m}\sum_{l=1}^{3}E_{ijl}\widehat{x}_{ijl}+\Delta}{\sum_{i=1}^{n_r}\sum_{j=1}^{m}\sum_{l=1}^{3}E_{ijl}x_{ijl}^{OPT}} \leq 3+\frac{\Delta}{E_{LP}^{(OPT)}}.\square$$

The parameter Δ and $E_{LP}^{(OPT)}$ in **Theorem 2** can be determined during the execution of *LP-HTA* algorithm.

Furthermore, the energy cost by transmission usually is much larger than that cost by computation, so that $E_{ij1} < E_{ij2} < E_{ij3}$ since more data are transmitted if we use base station or cloud to deal with task $\mathcal{T}_{ij}$. Therefore, we have the following corollary.

Corollary 1. *The ratio bound of the LP-HTA algorithm, denoted by R, satisfied that* $R \leq \min\{\frac{\max_{1\leq i\leq n_r,1\leq j\leq m}E_{ij3}}{\min_{1\leq i\leq n_r,1\leq j\leq m}E_{ij1}}, 3+\frac{\Delta}{E_{LP}^{(OPT)}}\}$*, on condition that* $E_{ij1} < E_{ij2} < E_{ij3}$.

Proof. Since $E_{ij1} < E_{ij2} < E_{ij3}$, we have $\sum_{i=1}^{n_r}\sum_{j=1}^{m}\sum_{l=1}^{3}E_{ijl}x_{ijl} \leq \sum_{i=1}^{n_r}\sum_{j=1}^{m}E_{ij3}$ and $\sum_{i=1}^{n_r}\sum_{j=1}^{m}\sum_{l=1}^{3}E_{ijl}x_{ijl}^{OPT} \geq \sum_{i=1}^{n_r}\sum_{j=1}^{m}E_{ij1}$, where $\{x_{ijl}|1 \leq i \leq n_r, 1 \leq j \leq m, l \in \{1,2,3\}\}$ is the assignment result return by *LP-HTA* algorithm.

Thus, $R \leq \frac{\max_{1\leq i\leq n_r,1\leq j\leq m}E_{ij3}}{\min_{1\leq i\leq n_r,1\leq j\leq m}E_{ij1}}$. That is, $R \leq \min\{\frac{\max_{1\leq i\leq n_r,1\leq j\leq m}E_{ij3}}{\min_{1\leq i\leq n_r,1\leq j\leq m}E_{ij1}}, 3+\frac{\Delta}{E_{LP}^{(OPT)}}\}$ based on Theorem 2. $\square$

Since $R \leq \min\{\frac{\max_{1\leq i\leq n_r,1\leq j\leq m}E_{ij3}}{\min_{1\leq i\leq n_r,1\leq j\leq m}E_{ij1}}, 3+\frac{\Delta}{E_{LP}^{(OPT)}}\} \leq \frac{\max_{1\leq i\leq n_r,1\leq j\leq m}E_{ij3}}{\min_{1\leq i\leq n_r,1\leq j\leq m}E_{ij1}}$ according to Corollary 1, and $n_r m$ is quite large comparing with $\frac{\max_{1\leq i\leq n_r,1\leq j\leq m}E_{ij3}}{\min_{1\leq i\leq n_r,1\leq j\leq m}E_{ij1}}$, thus the ratio bound of *LP-HTA* algorithm can be regarded as a constant.

IV. Divisible Tasks Assignment Algorithm

Besides the holistic computation tasks, there also exist many divisible tasks in a MEC system. We call a task to be divisible if and only if it can be implemented distributedly, *i.e.* the final result can be obtained by aggregating the partial results. For example, some statistics calculations, such as *Sum* or *Count*, can be regarded as divisible tasks. Considering that the partial results are always much smaller than the raw data, therefore, it will save much energy if we process such tasks in distributed manner. Furthermore, it is also useful to protect privacy if the tasks are processed distributedly in a MEC system [18].

Let D_i be the data owned by mobile device i, and $\{\mathcal{T}_{ij}|1 \leq i \leq n, 1 \leq j \leq m\}$ denote the set of all the tasks in a MEC system. Then, $D = \bigcup_{1\leq i\leq n,1\leq j\leq m}(LD_{ij} \bigcup ED_{ij})$ denote the total data required to be processed, where $D \subseteq \bigcup_{i=1}^{n} D_i$ and $D_i \bigcap D_j$ may not be empty since the monitoring regions of two devices may overlap with each other.

In order to reduce the energy cost as much as possible, we should rearrange the tasks according to $\{D_i|1 \leq i \leq n\}$. The aim of such rearrangement is that the new tasks assigned to each mobile device only need to deal with the local data. To achieve such aim and avoid the raw data transmission, the following two problems need to be solved.

(1). For each $1 \leq i \leq n$, how to determine the dataset that mobile device i needs to process?

(2). How to arrange the tasks using the above result?

Obviously, the first problem is more critical for the task arrangement in a MEC system. The following two subsections will discuss it based on different optimization goals, The last one will provide a solution for the second problem.

A. Data Division for Balancing the Workload

To determine the dataset for each mobile device i ($1 \leq i \leq n$) to process, we need divide D into several disjoint subsets, and each mobile device only response for dealing with a subset. Firstly, we want to divide D as uniform as possible due to two reasons. First, the computation workload of each mobile device will be balanced if D is divided uniformly, so that the energy consumption of each mobile edge will be balanced as well. Second, since the mobile device can process the tasks in parallel manner, the longest time spent by a mobile device will be shorten effectively if D is divided uniformly, which results in a smaller total processing time of the whole system. Based on such motivation, the definition of the *Optimal Coverage of D with Smallest Set Size* is given as follows.

Definition 1. Given D, $\{D_i|1 \leq i \leq n\}$, $\mathcal{C}^{(S)} = \{C_1^{(S)}, C_2^{(S)}, ..., C_n^{(S)}\}$ is called as an *Optimal Coverage of D with Smallest Set Size* if and only if $\mathcal{C}^{(S)}$ satisfies the following conditions: **(1)** $C_i^{(S)} \subset D \bigcap D_i$; **(2)** $C_i^{(S)} \bigcap C_j^{(S)} = \emptyset$ for any $1 \leq i \neq j \leq n$, and $\bigcup_{i=1}^{n} C_i^{(S)} = D$; and **(3)** $\max_{1\leq i\leq n}|C_i^{(S)}|$ is minimized.

The first condition of Definition 1 means that $C_i^{(S)}$ can be processed by mobile device i locally since $C_i^{(S)} \subseteq D_i$, and mobile device i need not to deal with any data if $C_i^{(S)} = \emptyset$. The second condition indicates that D is fully processed since $\bigcup_{i=1}^{n} C_i^{(S)} = D$, meanwhile, the redundant computation is avoided as $C_i^{(S)} \bigcap C_j^{(S)} = \emptyset$. The last condition guarantees that the processing time has been reduced as much as possible since $\max_{1\leq i\leq n}|C_i^{(S)}|$ is minimized.

Let $D = \{d_1, d_2, ..., d_M\}$ and $UD_i = D \bigcap D_i$ for all $1 \leq i \leq n$, where d_i is regarded as a data item or a data block determined by [19]. For $1 \leq r \leq M, 1 \leq i \leq n$, let y_{ri} be a indicator variable satisfies that $y_{ri} = 1$ if and only if d_r is assigned to mobile device i for processing, and p_{ri} denote the cost of such assignment. Since we want to avoid the raw data transmission, $p_{ri} = \infty$ if $d_r \notin UD_i$, otherwise $p_{ri} = 1$. Thus, the problem of calculating $\mathcal{C}^{(S)} = \{C_1^{(S)}, C_2^{(S)}, ..., C_n^{(S)}\}$, can be formalized as the following 0-1 programming problem.

$$\begin{aligned} P3: \min\ & maxsize \\ s.t.\ & \sum_{i=1}^{n} y_{ri} = 1, && 1 \leq r \leq M, \\ & \sum_{r=1}^{M} y_{ri}p_{ri} \leq maxsize, && 1 \leq r \leq N, \\ & y_{ri} \in \{0,1\}, && 1 \leq r \leq M, 1 \leq i \leq n. \end{aligned}$$

where $C_i = \{d_r | y_{ri} = 1\}$ for all $1 \leq i \leq n$.

Since 0-1 programming problem is NP-complete [20], it is hard to determine $\mathcal{C}^{(\mathcal{S})} = \{C_1^{(S)}, C_2^{(S)}, ..., C_n^{(S)}\}$ in polynomial time. Then, a greedy algorithm with $O(n)$ complexity is presented as follows. The algorithm contains three steps.

Step 1. Let $C_1^{(S)} = \emptyset$, $C_2^{(S)} = \emptyset$,..., $C_n^{(S)} = \emptyset$.

Step 2. Determine r by $r = \text{argmin}_{1 \leq i \leq n} |UD_i \cap D|$, set $C_r^{(S)} = UD_r \bigcap D$, and $D = D - C_r^{(S)}$.

Step 3. Repeat Step 2 until $D = \emptyset$. Return $\mathcal{C}^{(\mathcal{S})} = \{C_1^{(S)}, C_2^{(S)}, ..., C_n^{(S)}\}$.

The following theorem and corollary guarantee that the ratio bound of the above greedy algorithm is $\frac{1}{1-e^{-1}}$.

Theorem 3. *$f(\mathcal{X}) = \max_{A \in \mathcal{X}} |A|$ is a submodular function, where $\mathcal{X} \subseteq 2^D$, and $A \subseteq D$.*

Proof. Let $\mathcal{X} \subseteq 2^D$ and $\mathcal{Y} \subseteq 2^D$ be two sets satisfy that $\mathcal{X} \subseteq \mathcal{Y}$. Thus, we have $f(\mathcal{X}) = \max_{A \in \mathcal{X}} |A| \leq \max_{B \in \mathcal{Y}} |B| = f(\mathcal{Y})$.

Let $G \subseteq D$ and $G \notin \mathcal{Y}$. Therefore, there exists three cases that needs to be discussed.

Case 1. If there at least exists $\exists A \in \mathcal{X}$ satisfies that $A| > |G|$, then $f(\mathcal{X} \cup \{G\}) - f(\mathcal{X}) = 0$. Since $\mathcal{X} \subseteq \mathcal{Y}$, we have $A \in \mathcal{Y}$, and thus $f(\mathcal{X} \cup \{G\}) - f(\mathcal{X}) = 0 = f(\mathcal{Y} \cup \{G\}) - f(\mathcal{Y})$

Case 2. If $|G| > \max_{A \in \mathcal{X}} |A| (= f(\mathcal{X}))$, however, $\exists B \in \mathcal{Y} - \mathcal{X}$ satisfies that $|B| > |G|$, then we have $f(\mathcal{X} \cup \{G\}) - f(\mathcal{X}) = |G| - f(\mathcal{X}) > 0 = f(\mathcal{Y} \cup \{G\}) - f(\mathcal{Y})$

Case 3. If $|G| > \max_{B \in \mathcal{Y}} |B| (= f(\mathcal{Y})$, then $f(\mathcal{X} \cup \{G\}) - f(\mathcal{X}) = |G| - f(\mathcal{X}) \geq |G| - f(\mathcal{Y}) = f(\mathcal{Y} \cup \{G\}) - f(\mathcal{Y})$ since $f(\mathcal{X}) \leq f(\mathcal{Y})$

In summary, we have $f(\mathcal{X} \cup \{G\}) - f(\mathcal{X}) \geq f(\mathcal{Y} \cup \{G\}) - f(\mathcal{Y})$, so that $f(\mathcal{X})$ is a submodular function. □

Since $f(\mathcal{X})$ is a submodular function, the following Corollary will be obtained.

Corollary 2. *The ratio bound of greedy algorithm is $\frac{1}{1-e^{-1}}$.*

B. Data Division for Minizing Involved Mobile Devices

Secondly, in order to save energy for majority mobile devices, some applications may require to minimize the number of devices for dealing with the tasks. Based on such motivation, the *Optimal Coverage of D with Smallest Set Number* is defined as follows.

Definition 2. $\mathcal{C}^{(\mathcal{N})} = \{C_{l_1}^{(N)}, C_{l_2}^{(N)}, ..., C_{l_r}^{(N)}\}$ is called as an *Optimal Coverage of D with Smallest Set Number* if and only if $\mathcal{C}$ satisfies: **(1)** $C_{l_i}^{(N)} \neq \emptyset$, $C_{l_i}^{(N)} \subset D \bigcap D_{l_i}$; **(2)** $C_{l_i}^{(N)} \bigcap C_{l_j}^{(N)} = \emptyset$ for any $1 \leq i \neq j \leq n$, and $\bigcup_{i=1}^{l} C_{l_i}^{(N)} = D$; **(3)** For all $\mathcal{C}'$ which satisfies the above two conditions, we have $|\mathcal{C}'| \geq |\mathcal{C}^{(\mathcal{N})}| (= r)$.

Considering D is a universe, and $\mathcal{S}^{(\mathcal{N})} = \{UD_1, UD_2, ..., UD_n\}$ is a family of subsets of D, where $UD_i = D \bigcap D_i$ as given in Section IV.A. Therefore, the problem of calculating *Optimal Coverage of D with Smallest Set Number*, *i.e.* $\mathcal{C}^{(\mathcal{N})} = \{C_{l_1}^{(N)}, C_{l_2}^{(N)}, ..., C_{l_r}^{(N)}\}$, can be regarded as the Set Cover Problem with input D and $\mathcal{S}^{(\mathcal{N})}$. Since the Set Cover Problem is proved to be NP-complete in [21], therefore, it is hard to obtain $\mathcal{C}^{(\mathcal{N})} = \{C_{l_1}^{(N)}, C_{l_2}^{(N)}, ..., C_{l_r}^{(N)}\}$ in polynomial time. The following part will provide a greedy algorithm to get approximation result The algorithm consists of three steps, and whose pseudocode is given in Algorithm 1.

First, let $UD_i = D \bigcap D_i$ for all $1 \leq i \leq n$, and $\mathcal{C}^{(\mathcal{N})} = \emptyset$.

Second, determine r by $r = \text{argmax}_{1 \leq i \leq n} |UD_i \cap D|$, let $\mathcal{C}^{(\mathcal{N})} = \mathcal{C}^{(\mathcal{N})} \bigcup \{UD_r \cap D\}$ and $D = D - (UD_r \cap D)$;

Third, repeat Step 2 until $D = \emptyset$. Return $\mathcal{C}^{(\mathcal{N})}$.

Obversely, the complexity of the above algorithm is $O(n)$. Its ration bound is $O(\ln n)$.

C. Task Rearrangement Method

After obtaining $\mathcal{C}^{(\mathcal{S})} = \{C_1^{(S)}, C_2^{(S)}, ..., C_l^{(S)}\}$ or $\mathcal{C}^{(\mathcal{N})} = \{C_1^{(N)}, C_2^{(N)}, ..., C_l^{(N)}\}$, the computation tasks, $\{T_{ij} | 1 \leq i \leq n, 1 \leq j \leq m\}$ will be rearranged based on them.

Take $\mathcal{C}^{(\mathcal{S})} = \{C_1^{(S)}, C_2^{(S)}, ..., C_l^{(S)}\}$ as an example. For each device i, the dataset that it needs to process is $C_i^{(S)}$, therefore, the information of the task $\mathcal{T}_{rl}$, including op_{rl}, C_{rl} and T_{rl}, are transmit to mobile device i if $C_i^{(S)} \bigcap (LD_{ij} \bigcup ED_{ij}) \neq \emptyset$. Through such arrangement, every mobile devices are assigned a series of new computation tasks. Then, the *LP-HTA* algorithm in Section III is applied to schedule these new tasks and obtain the partial results. Finally, the partial results are aggregated according to users' requirements.

As only the task information and partial results are required to transmit in a MEC system, much energy will be saved.

V. Experiment Results

A. Experiment Settings

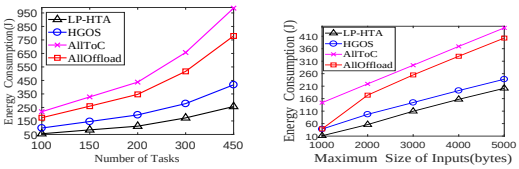
Similar as [22], we also assume that the energy cost, required CPU cycles and result size is linear to the size of input data in the simulated MEC system. That is, the required CPU cycles, the energy cost, and the result size can be estimated by λX, $\kappa \lambda X$ and ηX when the size of input data is X, where $\kappa = 10^{-27}$ (J/cycle), $\lambda = 330$ (cycle/byte), $\eta = 0.2$ based on [22]. The CPU frequencies of mobile devices varies from 1GHz to 2GHz. For a based station, its CPU frequency is set to be 4GHz, and the transmission delay between two base stations are 15ms [15]. For the cloud, we take Amazon T2.nano as example, and set its CPU frequency to be 2.4GHz, the delay between a base station and Cloud is 250ms based on [16]. Finally, each mobile device connects with the base station by 4G or WiFi randomly, where the download, upload, transmitting and receiving powers of simulated wireless network are summarized in Table I based on [23] and [24].

TABLE I: parameters of wireless networks

NetWork	Download speed	Upload speed	P^T	P^R
4G	13.76 Mbps	5.85 Mbps	7.32 W	1.6W
Wi-Fi	54.97 Mbps	12.88 Mbps	15.7 W	2.7 W

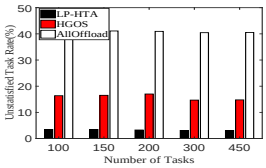
B. Performance of LP-HTA Algorithm

In this section, we will investigate the performance of our *LP-HTA* algorithm by comparing with *HGOS* and two baseline algorithms, *i.e. AllToC* and *AllOffload*, where *HGOS* denotes the Heuristic Greedy Offloading Scheme proposed in [12],



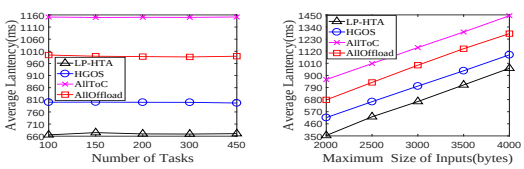
(a) Affected by Number of Tasks (b) Affected by Size of Input Data

Fig. 2: Energy Cost of *LP-HTA*



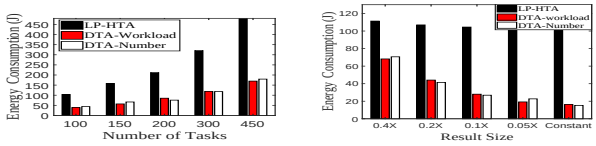
(a) Unsatisfied Task Rate

Fig. 3: The Rate of Unstatisfied Task



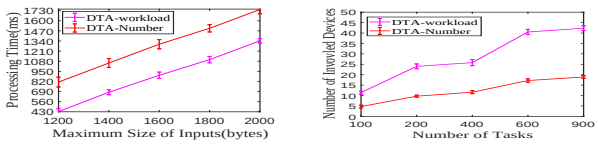
(a) Affected by Number of Tasks (b) Affected by Size of Input Data

Fig. 4: Latency of *LP-HTA*



(a) Affected by Number of Tasks (b) Affected by Size of Task Result

Fig. 5: Energy Cost of *LP-HTA*, *DTA-Workload*, *DTA-Number*



(a) Processing Time (b) Number of Invovled Devices

Fig. 6: The comparison of *DTA-Workload* and *DTA-Number*

AllToC and *AllOffload* means that all the tasks are offloaded to the cloud and to the base stations and the cloud, respectively. The reasons of choosing the above three methods are as follows. Firstly, we compare *LP-HTA* with *HGOS* [12] since it is the latest and efficient task assignment algorithms in a MEC system. Secondly, *AllToC* and *AllOffload* are selection for comparison because they are most classical and representative methods when the computation abilities of the mobile devices are not considered. The size of the external data is set to 0 to 0.5 times the local data.

The first group of experiments investigate the impact of the number of tasks on the energy cost of *LP-HTA*. In the experiments, the energy consumed by *LP-HTA*, *HGOS*, *AllToC* and *AllOffload* were calculated while the number of the tasks in a MEC system increased from 100 to 450, and the maximum input data size for each task is 3000kb. According to the results illustrated in Fig.2(a), the energy consumption of *LP-HTA* is much smaller than that of *AllToC* and *AllOffload* as the computation abilities of the mobile edges, including mobile devices and the base stations are sufficiently used. Furthermore, the energy consumed by our *LP-HTA* is also lower than that consumed by *HGOS*. Since the constraints of each mobile edge is fully considered during task offloading, the task assignment results of our *LP-HTA* is more reasonable and efficient comparing with *HGOS*, and thus much energy will be saved. Finally, we find that the energy consumed by *LP-HTA* increases slowly with the increment of the tasks, which means that our *LP-HTA* algorithm is very suitable to process the computation intensive tasks.

The second group of experiments observe the energy consumption of *LP-HTA* affected by the input data size. In the experiments, the energy cost of *LP-HTA*, *HGOS*,*AllToC* and *AllOffload* were calculated while the maximum input data size of each task increased form 1000 kb to 5000 kb, and the number of total tasks is 100. The results presented in Fig.2(b) indicate that the energy consumption of *LP-HTA* is also the smallest even when the amount of data required to be processed increases, which means that our *LP-HTA* is also suitable to deal with the data-intensive tasks.

The third group of experiments is to investigate the unsatisfied task rate of all the task assignment algorithms. The unsatisfied task rate is denoted by the proportion of the tasks whose delay constraints can not be met among all the tasks. Since the unsatisfied task rate of *AllToC* is quite high due to the large latencies of transmitting all the task to the remote cloud, we only compare *HGOS*, *AllOffload* and our *LP-HTA* in the experiments. As shows in Fig.3, the unsatisfied task rate of the three algorithms were calculated when the number of tasks increased form 100 to 450. The results shows that the unsatisfied task rate of our *LP-HTA* is quite small comparing with *HGOS* and *AllOffload*. Since we take the delay constraint of each task into account during task assignment, only a small portion of task are unsatisfied. Therefore, our *LP-HTA* algorithms are more efficient and effective to process the tasks that has strict delay constraints. Furthermore, although the energy cost of *HGOS* are close to *LP-HTA* based on Fig.2, it has quite large unsatisfied task rate, so that the performance of *HGOS* is not good enough comparing with *LP-HTA* as large amount of deadline constraints cannot be met.

The fourth group of experiments is to investigate the impact of the amount of tasks on the latency of *LP-HTA*. In the experiments, the average latency of *LP-HTA*, *HGOS*, *AllToC* and *AllOffload* were calculated while the number of tasks grows from 100 to 450, and the maximum size of input data is 3000kb. From results in Fig.4(a), we can see that the average latency of *LP-HTA* is extremely small comparing with *AllToC* and *AllOffload* since the mobile devices are fully used and some of task are processed immediately by *LP-HTA*. Moreover, the average latency of *LP-HTA* is also much lower than that of *HGOS* as *LP-HTA* considers both of task deadline constraints and the limited computing resources of each mobile edge sufficiently during task assignment. Thus,LP-HTA is efficient to process the latency-intensive tasks in MEC systems.

The fifth group of experiments is used to analyze the

relationship between the latency of *LP-HTA* and the size of input data. The average latency of *LP-HTA*, *HGOS*, *AllToC* and *AllOffload* were calculated while the maximum size of the input data for each task varies from 1000kb to 5000kb, and the number of total tasks is 100. Based on the results in Fig.4(b), the average latency of *LP-HTA* is still the smallest comparing with *HGOS*, *AllToC* and *AllOffload*. However, the advantage of *LP-HTA* on latency is not so much obvious comparing with *HGOS*, which is because more tasks will exceed the processing abilities of mobile devices with the increment of input data size for both *LP-HTA* and *HGOS*, so that they are offloaded to base station or to the cloud, and the latency will enlarge accordingly. In spite of that, the *LP-HTA* still utilize the processing abilities of the mobile edges, including mobile devices and base stations, as much as possible. Meanwhile, the *LP-HTA* also guarantees that much more deadline constraints of the tasks are satisfied even when the average latency is enlarged according to the results in Fig.3. Such results also verified that our *LP-HTA* is efficient and effective to deal with the data-intensive tasks.

C. Performance of DTA

The section will evaluate of the performance of Divisible Task Assignment algorithms, and we use *DTA-Workload* and *DTA-Number* to stand for the algorithms proposed in Sections IV(a) and IV(b), respectively. In the following experiments, the comparisons among *DTA-Workload*, *DTA-Number* and *LP-HTA* are considered since *LP-HTA* has the best performance according to the analysis in the above section

The first group of experiments compares the energy cost by *DTA-Workload*,*DTA-Number* and *LP-HTA* for processing the tasks with different amount. The energy cost of three methods were computed while the number of tasks increased form 100 to 450, the maximum size of input data was 3000kb for each task, and the ratio of the result size to the amount of input data was 0.2. Fig.5(a) shows that the energy cost of *DTA-Workload* and *DTA-Number* are quite small especially when the amount of tasks increases. Since more raw data are avoided to transmit by *DTA-Workload* and *DTA-Number* when the amount of tasks increases, lots of energy will be saved.

The second group of experiments observes the energy cost of the above three algorithms while the result size is varying. In the experiments, the result size is set to be $0.4X$, $0.2X$, $0.1X$, $0.05X$ and constant, where X is the size of input data. The number of total tasks in a MEC system was 100, and the maximum size of input data was 3000kb for each task. Fig.5(a) presents the energy cost of three algorithm for different result size. It shows that *DTA-Workload* and *DTA-Number* consume extremely small energy when the result size decreases. Considering that the sizes of final result and partial results are alway much smaller than that of raw data, *DTA-Workload* and *DTA-Number* will achieve high performance for processing many kinds of tasks.

The last two groups of experiments are going to compare *DTA-Workload* and *DTA-Number* separately. In Fig.6(a), the processing time of *DTA-Workload* and *DTA-Number* were calculated while the maximum size of input data increased from 1200kb to 2000kb, and the number of total tasks was 200. In Fig.6(b), the numbers of involved mobile devices for *DTA-Workload* and *DTA-Number* were counted while the number of total tasks varied form 100 to 900, and the maximum size of input data was 2000kb for each task. The results in Fig.6(a) and Fig.6(b) show that the processing time of *DTA-Workload* is much smaller since it divides the input data as uniform as possible, and the total processing workload are balanced, so that the processing time is shorten accordingly. On the other hand, smaller amount of mobile devices are required by *DTA-Number*, so that the energy of majority mobile devices is save. Therefore, *DTA-Workload* and *DTA-Number* are suitable for different situations, where the users can select one of them flexibly according to the applications.

VI. Related Works

The early ones that studied the task assignment in a MEC system mainly focus on serving a single user. To determine the offloading strategies of the tasks proposed by single users, a Binary offloading algorithms is given in [6]. To take advantage of parallel computing, a partial offloading problem were studied in [25], and an approximation algorithm with $(1+\epsilon)$ ratio bound was given to solve it. In [26], a variable-substitution technique was provided to jointly optimize the offloading ratio, transmission power and CPU-cycle frequency. These works has high performance to deal with the tasks raised by a single user, however, the situation considered by them seems a little simple. Meanwhile, they did not consider the data distribution during determine the offloading strategies.

Considering that a MEC system always serve for multiple users, [8] has formulated the computation offloading problem of multiple users via a single wireless access point as a computation offloading game, and proposed a decentralized mechanism that can achieve the Nash equilibrium to solve it. In [27], the author try to optimize the uplink and downlink scheduling using queuing theory. In [28], a decomposition algorithm was given to control the computation offloading selection, clock frequency control and transmission power allocation iteratively. The above algorithms are efficient to process the tasks raised by multiple users, however, since they were designed for a MEC system with a single base station, the cooperations among different base stations were not considered sufficiently. Furthermore, these algorithms ignored the data distribution either during task assignment. Besides, [11] introduced the centralized and distributed Greedy Maximal Scheduling algorithms to solve the computation offloading problem for multiple users with multiple tasks. However, these algorithms did not consider the data sharing during the task assignment, and ignored the delay constraint of each task as well, so that they are not suitable to solve the problem discussed in this paper.

To enhance the cooperations among different base stations, a new cooperation scheme among base stations was given in [29], the author tried to reduce the response latency through caching the results of the popular computation tasks. [30]

proposed a game-theoretic algorithm to maximize revenues from all applications. All above works fully utilized the cooperations among base stations. However, these algorithms ignored the distribution of input data as well, and they are not suitable to the Data-Shared MEC systems. [31] provided a payment-based incentive mechanism. Such mechanism supports sharing computation resources among base stations and the authors also established a social trust network to manage the security risks among them, however, it still has the above problems that we pointed out. Moreover, the constraints on processing ability of each base station are not reasonable enough since they only consider the task arrival rate and didn't take the data and resources occupied by each task into account.

More recently, the architecture of a MEC system which involves the three levels and serves for multiple users are sufficiently investigated by [12] and [13]. In [12], a heuristic greedy offloading scheme was given for ultra dense networks. In [13], a distributed computation offloading algorithm that can achieve the Nash equilibrium was provided. Both of these works are efficient for a MEC system, however, they did not considered data distribution and the delay constraints of tasks during task assignment, thus they are not able to assign the tasks in a Data-Shared MEC system as well. Besides, the processing abilities of the Cloud are not fully utilized, either.

VII. Conclusion

This paper studies the task assignment problem in a Data-Share MEC system. We firstly distinguish the computation tasks as the holistic tasks and the divisible tasks. For the holistic tasks, the HTA problem is proposed and proved to be NP-complete. Finally, a linear programming based approximation algorithm is proposed. For the divisible tasks, the key problem is to rearrange the tasks according to the data distribution. Two approximation algorithms with $O(n)$ complexities are provided to solve it under different optimization goals. Both of theoretical analysis and experiments show that all proposed algorithms achieve high performance.

Acknowledgments

This work is supported by the National Natural Science Foundation of China under Grant Nos. 61632010, U1509216.

References

[1] S. Cheng, Y. Li, Z. Tian, W. Cheng, and X. Cheng, "A model for integrating heterogeneous sensory data in iot systems," *Computer Networks*, vol. 150, pp. 1–14, 2019.

[2] S. Cheng, Z. Cai, and J. Li, "Curve query processing in wireless sensor networks," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 11, pp. 5198–5209, 2015.

[3] S. Cheng, Z. Cai, J. Li, and H. Gao, "Extracting kernel dataset from big sensory data in wireless sensor networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 4, pp. 813–827, 2017.

[4] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computingła key technology towards 5g," *ETSI white paper*, vol. 11, no. 11, pp. 1–16, 2015.

[5] A. Ahmed and E. Ahmed, "A survey on mobile edge computing," in *International Conference on Intelligent Systems and Control*, 2016.

[6] W. Zhang, Y. Wen, K. Guan, K. Dan, H. Luo, and D. O. Wu, "Energy-optimal mobile cloud computing under stochastic wireless channel," *IEEE TWC*, vol. 12, no. 9, pp. 4569–4581, 2013.

[7] K. Kumar, J. Liu, Y. H. Lu, and B. Bhargava, "A survey of computation offloading for mobile systems," *Mobile Networks & Applications*, vol. 18, no. 1, pp. 129–140, 2013.

[8] X. Chen, "Decentralized computation offloading game for mobile cloud computing," *Parallel & Distributed Systems IEEE Transactions on*, vol. 26, no. 4, pp. 974–983, 2014.

[9] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Transactions on Networking*, vol. 24, no. 5, pp. 2795–2808, 2016.

[10] T. Q. Dinh, J. Tang, Q. D. La, and T. Q. S. Quek, "Offloading in mobile edge computing: Task allocation and computational frequency scaling," *IEEE TCOM*, vol. 65, no. 8, pp. 3571–3584, 2017.

[11] W. Chen, D. Wang, and K. Li, "Multi-user multi-task computation offloading in green mobile edge cloud computing," *IEEE Transactions on Services Computing*, 2018.

[12] H. Guo, J. Liu, and J. Zhang, "Computation offloading for multi-access mobile edge computing in ultra-dense networks," *IEEE Communications Magazine*, vol. 56, no. 8, pp. 14–19, 2018.

[13] L. Tang and S. He, "Multi-user computation offloading in mobile edge computing: A behavioral perspective," *IEEE Network*, vol. 32, no. 1, pp. 48–53, 2018.

[14] T. D. Burd and R. W. Brodersen, *Processor design for portable systems*. Kluwer Academic Publishers, 1996.

[15] J. Oueis, E. Calvanese-Strinati, A. De Domenico, and S. Barbarossa, "On the impact of backhaul network on distributed cloud computing," in *Wireless Communications and NETWORKING Conference Workshops*, 2014, pp. 12–17.

[16] "Amazon Cloud." [Online]. Available: http://www.cloudping.info/

[17] N. Karmarkar, "A new polynomial-time algorithm for linear programming," *Combinatorica*, vol. 4, no. 4, pp. 373–395, 1984.

[18] R. Gennaro, C. Gentry, and B. Parno, "Non-interactive verifiable computing: Outsourcing computation to untrusted workers," in *Annual Cryptology Conference*, 2010, pp. 465–482.

[19] Y. Huang, X. Song, Y. Fan, Y. Yang, and X. Li, "Fair caching algorithms for peer data sharing in pervasive edge computing environments," in *ICDCS*, 2017.

[20] C. H. Papadimitriou, "On the complexity of integer programming," *Journal of the Acm*, vol. 28, no. 4, pp. 765–768, 1981.

[21] U. Feige, *A threshold of ln n for approximating set cover*. ACM, 1998.

[22] O. Munoz, A. Pascual-Iserte, and J. Vidal, "Optimization of radio and computational resources for energy efficiency in latency-constrained application offloading," *IEEE Transactions on Vehicular Technology*, vol. 64, no. 10, pp. 4738–4755, 2015.

[23] A. E. Eshratifar and M. Pedram, "Energy and performance efficient computation offloading for deep neural networks in a mobile cloud computing environment," in *Proceedings of the 2018 on Great Lakes Symposium on VLSI*, 2018, pp. 111–116.

[24] J. Huang, F. Qian, A. Gerber, Z. M. Mao, S. Sen, and O. Spatscheck, "A close examination of performance and power characteristics of 4g lte networks," in *ACM Mobisys*, 2012, pp. 225–238.

[25] Y. H. Kao, B. Krishnamachari, M. R. Ra, and B. Fan, "Hermes: Latency optimal task assignment for resource-constrained mobile computing," in *IEEE Conference on Computer Communications*, 2015, pp. 1894–1902.

[26] Y. Wang, M. Sheng, X. Wang, L. Wang, and J. Li, "Mobile-edge computing: Partial computation offloading using dynamic voltage scaling," *IEEE TCOM*, vol. 64, no. 10, pp. 4268–4282, 2016.

[27] M. Molina, O. Muoz, A. Pascual-Iserte, and J. Vidal, "Joint scheduling of communication and computation resources in multiuser wireless application offloading," in *IEEE International Symposium on Personal, Indoor, and Mobile Radio Communication*, 2015, pp. 1093–1098.

[28] S. Guo, B. Xiao, Y. Yang, and Y. Yang, "Energy-efficient dynamic offloading and resource scheduling in mobile cloud computing," in *INFOCOM*, 2016, pp. 1–9.

[29] M. S. Elbamby, M. Bennis, and W. Saad, "Proactive edge computing in latency-constrained fog networks," in *European Conference on Networks and Communications*, 2017, pp. 1–6.

[30] R. Yu, J. Ding, S. Maharjan, S. Gjessing, Y. Zhang, and D. H. K. Tsang, "Decentralized and optimal resource cooperation in geo-distributed mobile cloud computing," *IEEE Transactions on Emerging Topics in Computing*, vol. PP, no. 99, pp. 1–1, 2018.

[31] L. Chen and J. Xu, "Socially trusted collaborative edge computing in ultra dense networks," in *Proceedings of the Second ACM/IEEE Symposium on Edge Computing*, 2017, p. 9.