Activate Cost-Effective Mobile Crowd Sensing with Multi-access Edge Computing

Zhengzhe Xiang[1], Shuiguang Deng[2], Yuhang Zheng[2], Dongjing Wang[3], Cheng Zhang[2], Yuanyi Chen[1], and Zengwei Zheng[1](✉)

[1] Zhejiang University City College, Hangzhou, China
{xiangzz,chenyuanyi,zhengzw}@zucc.edu.cn
[2] Computer College, Zhejiang University, Hangzhou, China
{dengsg,zyhxxds_ludwig,coolzc}@zju.edu.cn
[3] School of Computer Science and Technology, Hangzhou Dianzi University, Hangzhou, China
Dongjing.Wang@hdu.edu.cn

Abstract. Recently, the mobile crowd sensing (MCS) technique is believed to be an important role in multi-source data acquisition tasks. With devices or people with different sensing abilities in the cities, we can easily split and distribute the complex task in an appropriate way so that those devices or people can be stimulated to collect data within different scopes individually, while the results of them can be analyzed and integrated collaboratively to fulfill that complex task. However, in typical centralized architecture, the latency brought by unstable and time-consuming long-distance network transmission limits the development of MCS. The multi-access edge computing (MEC) technique is now regarded as the key tool to solve this problem. By establishing a service provisioning system based at the edge of the network, the latency can be reduced and the analysis or integration can also be conducted in time with the help of corresponding services deployed on nearby edge servers. However, as the edge servers are resource-limited, the sensing abilities vary among devices or people, and the budget of fulfilling a task is determined, we should be more careful in task assignment and service deployment. In this paper, we investigate the relationship between the task quality and the cost in the MEC-based MCS system and propose the analysis framework of it based on two classical cost-performance balancing problems. Besides, we conduct comprehensive experiments to evaluate the performance of our approach. The results show that the proposed approach can easily obtain exact solutions, and the factors that may impact the results are also adequately explored.

Keywords: Multi-access Edge Computing · Mobile crowd sensing · Task assignment · Service deployment · Incentive mechanism

H. Gao et al. (Eds.): ChinaCom 2020, LNICST 352, pp. 78–97, 2021.
https://doi.org/10.1007/978-3-030-67720-6_6

1 Introduction

With the development of mobile computing technology, we are now embracing an era of mobile devices and services [1]. According to the report of GSMA[1], about 5.1 billion people around the world have subscribed to mobile applications, and the number will increase at an average annual growth rate of 1.9% before 2025. As a result, mobile devices and mobile applications are becoming increasingly important and remolded the communication between people and machines. The tremendous increasing number of mobile users and devices has created a huge market that draws the attention of all the world. To make themselves to be the best one among the competitors, mobile application enterprises all want to better understand the preferences of these users and discover their underlying behavior patterns. Therefore, the researchers of these enterprises will always try their best to collect users' behavior records or even interview their target users directly, because they are sure that these structured/unstructured and sequential/non-sequential context data will help them to build a general user portrait model to analyze and predict users' future behaviors. However, as people are rarely willing to provide their data because of their subconscious privacy protection and the worry about the energy consumption of the external computation, it will be hard for the application developers to collect enough high-quality data for their Artificial Intelligence (AI) models legally. To solve this problem, more and more developers turn to the mobile crowd sensing (MCS) technique. Specifically, MCS is a human-oriented technique that leverages the built-in sensors of users' mobile devices as well as the involvement of users to collect data. It does not only care about the effectiveness and accuracy of the data but also focuses on the issues of stimulating users to share their data. With the MCS technique, a reliable publish/subscribe interaction framework is established between users and developers so that high-quality data can be collected with the admissions and willingness of the users if the developers can pay for their cooperation. However, the latency brought by long-distance transmission and traffic congestion of huge data in the network, as well as the energy consumption brought by data pre-processing limits the applications of MCS in typical centralized architecture.

Fortunately, Multi-access Edge Computing (MEC) technique is proposed to solve the aforementioned problems [2–4]. MEC is a novel paradigm that has emerged recently as a reinforcement of mobile cloud computing, to optimize the mobile resource usage and wireless network to provide context-aware services [5,6]. With the help of MEC, computation and transmission between mobile devices and the cloud are partly migrated to edge servers. Therefore, users can easily connect to their nearby edge servers via wireless network [7] and offload their tasks to them. The short-distance connection between users and edge servers can dramatically reduce the latency, and the computation capability of the edge servers are quite qualified to finish those conventional tasks. What's more, with the help of the container platforms in the limelight like **Kubernetes**, it will be easy to manage services (e.g. the data pre-processing services) in the

[1] https://www.gsmaintelligence.com/.

MEC environment. However, these advantages cannot be the causes of the carelessness in planning the multi-source data acquisition—if the sensing tasks are not assigned to appropriate users, the data acquisition task may even obtain lower-quality data with much higher cost. More critically, as the edge servers are all resource-constrained [8,9], if the data pre-processing services are not deployed on appropriate edge servers, there would be no enough resources for them to run. Thus, it will be important to design a task assignment scheme as well as a service deployment scheme to balance the quality and cost. The main contributions of this paper are:

1. We investigate the relationship between the task quality and the cost in the MEC-based MCS problem and propose the analysis framework of it based on two classical cost-performance balancing problems.
2. We mathematically model the former problems which aim to balance the task quality and the cost under the constraints of application developers' budget, available resources of edge servers, and the capacities of users as a mixed-integer quadratic programming (MIQP) problem.
3. We conduct a series of experiments to evaluate the results of the solutions and show the improvement compared with other existing baselines. Besides, different configurations of the MEC-based MCS system are investigated to explore the impacts of related factors.

The rest of this paper is organized as follows. In Sect. 2, we use a simple example to show how the task assignment scheme and service development scheme can impact the quality and cost in the MEC-based MCS system. Section 3 presents definitions, concepts and components of the proposed problem. Section 4 describes the approaches we proposed to solve this problem. Section 5 shows the experimental results including the factors that affect our algorithms. Section 6 highlights the related work of the incentive mechanism and task assignment approaches. Finally, Sect. 7 concludes our contribution and outlines future work.

2 Motivation Scenario

In this section, we will briefly introduce the mechanism of how multi-access edge computing techniques can be used to help to optimize the mobile crowd sensing tasks by giving an example of a multi-source data acquisition task in customer portrait construction.

Figure 1 gives an overview of the West Lake business district in Hangzhou city. The wide variety of shops in this business district attract a large number of users every day, and the behavior records of these users in the business district will be a good data source for constructing customer portraits. Therefore, data researchers hope to use mobile crowd sensing technique to collect these data. Specifically, the data that is expected to be collected may include T_1) recent walking distance in the business district T_2) the most frequently visited shops T_3) recent bills in the business district. However, not all stimulated users are willing or able to provide all the data the researchers want, and even these

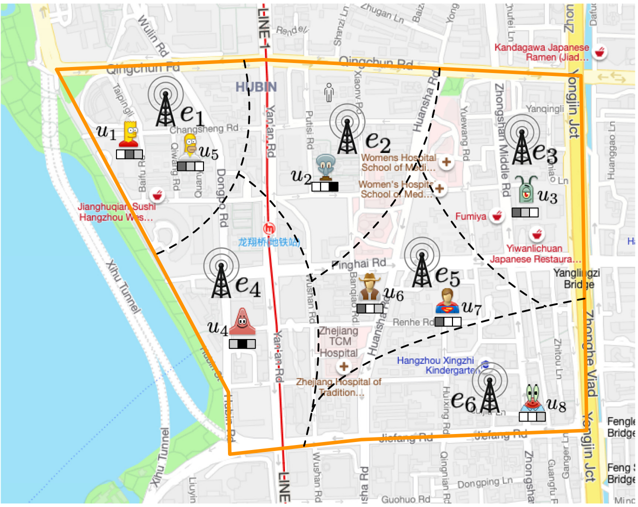

Fig. 1. A customer portrait example

users will have differences in the quality of the provided data. At this time, in order to construct a customer portrait that reflects the "average customer", the data researchers will assume that these customers are sampled from the same "customer space", and assign the three types of data collection tasks to different users, and use the MEC server nearby to pre-process the collected data.

As shown in this figure, edge servers distributed in this business district have their own serving area so that the users can easily connect to the nearest one. There is a rectangle group under every user to describe their willingness for providing different types of data (T_1–T_3), and the user can provide data with better quality if his corresponding rectangles are darker in color. Suppose the users will be equally rewarded if they are assigned with one task. Therefore, we can let user u_6 and u_7 to provide data T_1, let user u_4 to provide data T_2 and let user u_2 to provide data T_3. Meanwhile, the pre-processing module of T_1 can be deployed on e_5, T_2 on e_4 and T_3 on e_2. It is obvious that we can move the pre-processing module for T_3 to e_5 and let u_6 to provide data, but the quality will get worse because user u_6 always pay in cash and the transaction data on his device is not complete.

3 System Model and Problem Description

Although the example in Sect. 2 has given a brief introduction about the scenario, more details like costs and capacities are ignored in it. Therefore, we will give a complete system model in this section and then describe the quality-cost balancing problem (Fig. 2).

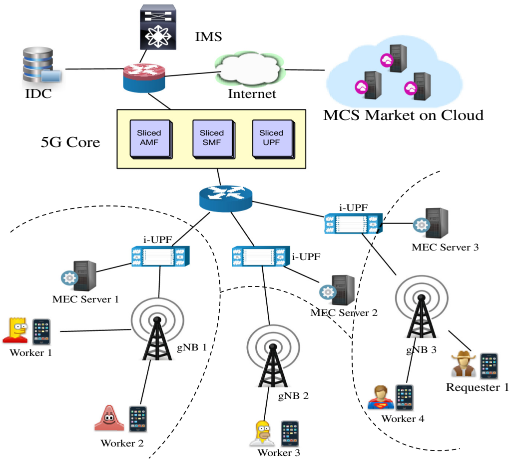

Fig. 2. An illustration of the MEC-based MCS system

3.1 System Entities

In more general MCS systems, the users who submit tasks are called *requesters*, and the users who fulfill the tasks with their mobile devices are called *workers*. Suppose $U = \{u_1, u_2, \ldots, u_M\}$ is the set of mobile users, and then every user can register as a requester or a worker. In this paper, we denote $\mathcal{E} = \{e_1, e_2, \ldots, e_N\}$ as the set of edge servers in the MEC-based MCS system. These edge servers have different serving areas in which the users can connect to them for receiving the sensing tasks. Without loss of generality, the principle of proximity is adopted here so that the users will communicate to their nearest edge server. If we use $\mathcal{L}_{m,j} \in \{0, 1\}$ as an indicator to describe the user coverage of edge servers (namely, whether user u_m is in the serving area of e_j or not), we have

$$\sum_{j=1}^{N} \mathcal{L}_{m,j} = 1, \forall m \in [1, M] \tag{1}$$

3.2 Task Announcement and Deployment

If a user chooses to register as a requester, he/she can then submit a new task T, e.g. a multi-source data acquisition task. When task T is submitted to the MCS market by requester, the MCS market will then decompose task T into several sensing sub-tasks $(T_1, T_2, \ldots, T_K)$. Every sensing sub-task T_k receives the sensing data from mobile devices and conducts a series of analyses to obtain the result that T_k is expected to acquire. For example, the submitted task T may be a multi-source data based machine learning task, and K different kinds of data are needed to be collected from various users or devices. With the help of MEC architecture, the feature pre-processing can be obtained with deployed related services of T_1–T_K on the edge servers in a distributed way. As web services are usually the carriers to fulfill corresponding tasks [10] in practice, we use $S = (s_1, s_2, \ldots, s_K)$ to denote them. These services can be launched as instances on edge servers to deal with the collected data about the sub-tasks. Here we use Q_k to describe how the task T_k is completed in the MEC-based MCS system with s_k, namely the completion quality. In general, there will be a minimum requirement ε_k for the completion quality for given task T_k to ensure that the result of T is satisfying. Because edge servers are resource-constrained, at most $n_j^{\mathcal{E}}$ service instances are allowed to be deployed on edge server e_j. Denote $D_{j,k}$ as the indicator to show whether the instance of s_k is deployed on edge server e_j or not, which is formally defined as:

$$D_{j,k} = \begin{cases} 1, s_k\text{'s instance is deployed on } e_j \\ 0, \text{otherwise} \end{cases}, \tag{2}$$

then the constraints of edge resource can be represented with:

$$\sum_{k=1}^{K} D_{j,k} \leq n_j^{\mathcal{E}}, \forall j \in [1, N] \tag{3}$$

3.3 Sub-task Assignment

If a user chooses to register as a worker, he/she then must claim his/her willing tasks via informing the MCS scheduler what kinds of sub-tasks he/she can complete:

$$\mathcal{W}_{m,k} = \begin{cases} 1, u_m \text{ would like to fulfill } T_k \\ 0, \text{otherwise} \end{cases} \tag{4}$$

Besides, the user also needs to tell the MCS scheduler his/her capacity to complete the tasks because the users can never be perpetual-motion machines. For example, the capacity may be the remaining battery of their devices when the sub-tasks are energy-consuming, and it may also be the left physical strength of them if the sub-tasks would exhaust people, like counting the building in a street. Suppose the capacity of user u_m is denoted with n_m^U, and $\mathcal{P}_{m,k}$ denotes how many times user u_m is arranged to complete T_k, then we have:

$$\sum_{k=1}^{K} \mathcal{P}_{m,k} \leq n_m^U, \forall m \in [1, M] \tag{5}$$

To evaluate the task completion quality of T and the total cost of the requester, here we use $q_{m,k}$ to denote the task completion quality of sub-task T_k under user u_m because the workers are heterogeneous, and use $c_{m,k}$ to denote the incentive requirement of u_m if he/she would like to complete the sub-task T_k. Then, the task completion quality of T_k can be represented with:

$$Q_k(\boldsymbol{\mathcal{P}}, \boldsymbol{\mathcal{D}}) = \sum_{j=1}^{N} \sum_{m=1}^{M} \mathcal{L}_{m,j} \cdot \mathcal{D}_{j,k} \cdot W_{m,k} \cdot q_{m,k} \cdot \mathcal{P}_{m,k} \tag{6}$$

and when the price of deploying a service instance is ν, the total cost of the requester can be represented with:

$$C_r(\boldsymbol{\mathcal{P}}, \boldsymbol{\mathcal{D}}) = \sum_{m=1}^{M} \sum_{k=1}^{K} \mathcal{P}_{m,k} \cdot c_{m,k} + \nu \sum_{j=1}^{N} \sum_{k=1}^{K} \mathcal{D}_{j,k} \tag{7}$$

3.4 Problem Definition and Formulation

With the introduction of related concepts, now we can give the definition of this **c**ost-effective **s**ervice **d**eployment **a**nd **t**ask **a**ssignment for **M**CS in **M**EC (CSDATAM$_2$) problem clearly. In this CSDATAM$_2$ problem, the requester would always like to balance the task completion quality and the cost to have his/her submitted task completed, so he/she would like to find the optimal service deployment strategy $\boldsymbol{\mathcal{P}}$ and task assignment strategy $\boldsymbol{\mathcal{P}}$ to make ends meet. Therefore, we can now formulate the CSDATAM$_2$ problem from the perspective of cost optimization (called CSDATAM$_2$-C):

$$P_C: \quad \min\ C_r(\boldsymbol{\mathcal{P}}, \boldsymbol{\mathcal{D}}) \tag{8}$$

$$s.t. \quad Q_k(\boldsymbol{\mathcal{P}}, \boldsymbol{\mathcal{D}}) \geq \varepsilon_k, \forall k \tag{9}$$

$$\sum_{k=1}^{K} \mathcal{D}_{j,k} \leq n_j^{\mathcal{E}}, \forall j \tag{10}$$

$$\sum_{k=1}^{K} \mathcal{P}_{m,k} \leq n_m^{U}, \forall m \tag{11}$$

$$\mathcal{D}_{j,k} \in \{0, 1\}, \forall j, \forall k \tag{12}$$

$$\mathcal{P}_{m,k} \in \mathbb{N}, \forall m, \forall k \tag{13}$$

In this problem, every sub-task T_k is required to be completed with a minimum task completion quality ε_k.

Similarly, from the perspective of task completion quality, we can also defined CSDATAM$_2$-Q problem as follows:

$$P_Q: \quad \max \sum_{k=1}^{K} Q_k(\boldsymbol{\mathcal{P}}, \boldsymbol{\mathcal{D}}) \tag{14}$$

$$s.t. \quad Q_k(\boldsymbol{\mathcal{P}}, \boldsymbol{\mathcal{D}}) \geq \varepsilon_k, \forall k \tag{15}$$

$$C_r(\boldsymbol{\mathcal{P}}, \boldsymbol{\mathcal{D}}) \leq C^{\star}, \tag{16}$$

$$\sum_{k=1}^{K} \mathcal{D}_{j,k} \leq n_j^{\mathcal{E}}, \forall j \tag{17}$$

$$\sum_{k=1}^{K} \mathcal{P}_{m,k} \leq n_m^{U}, \forall m \tag{18}$$

$$\mathcal{D}_{j,k} \in \{0, 1\}, \forall j, \forall k \tag{19}$$

$$\mathcal{P}_{m,k} \in \mathbb{N}, \forall m, \forall k \tag{20}$$

We can find that in this version, we mainly focus on the total task completion quality while keeping the total cost within an acceptable budget $C^{\star}$.

4 The Optimal Service Deployment and Task Assignment Strategies

In this section, we reform the representation the problem formulated in Sect. 3.4. First of all, as there are two decision variables $\boldsymbol{\mathcal{P}}$ and $\boldsymbol{\mathcal{D}}$ as well as other constant matrices involved in this problem, here we will vectorize them to obtain the simplified representation. For the decision variables, we denote

$$\boldsymbol{\mathcal{P}}_k = \begin{bmatrix} \mathcal{P}_{1,k} \\ \mathcal{P}_{2,k} \\ \vdots \\ \mathcal{P}_{M,k} \end{bmatrix}, \boldsymbol{\mathcal{D}}_k = \begin{bmatrix} \mathcal{D}_{1,k} \\ \mathcal{D}_{2,k} \\ \vdots \\ \mathcal{D}_{N,k} \end{bmatrix} \tag{21}$$

as the column vectors of the origin matrices, and denote $\boldsymbol{x} = [\boldsymbol{p}\, \boldsymbol{d}]^T$ as the unified decision where $\boldsymbol{p} = [\boldsymbol{\mathcal{P}_1}, \boldsymbol{\mathcal{P}_2}, ..., \boldsymbol{\mathcal{P}_K}]^T$ and $\boldsymbol{d} = [\boldsymbol{\mathcal{D}_1}, \boldsymbol{\mathcal{D}_2}, ..., \boldsymbol{\mathcal{D}_K}]^T$. At the same time, with the constant variables denoted with:

$$\boldsymbol{\mathcal{W}}_k = \begin{bmatrix} \mathcal{W}_{1,k} \\ \mathcal{W}_{2,k} \\ \vdots \\ \mathcal{W}_{M,k} \end{bmatrix}, \boldsymbol{q}_k = \begin{bmatrix} q_{1,k} \\ q_{2,k} \\ \vdots \\ q_{M,k} \end{bmatrix}, \boldsymbol{c}_k = \begin{bmatrix} c_{1,k} \\ c_{2,k} \\ \vdots \\ c_{M,k} \end{bmatrix} \tag{22}$$

and $\boldsymbol{c} = [\boldsymbol{c}_1, \boldsymbol{c}_1, ..., \boldsymbol{c}_K]^T, \boldsymbol{n}^{\mathcal{U}} = [n_1^{\mathcal{U}}, n_2^{\mathcal{U}}, ..., n_M^{\mathcal{U}}]^T, \boldsymbol{n}^{\mathcal{E}} = [n_1^{\mathcal{E}}, n_2^{\mathcal{E}}, ..., n_N^{\mathcal{E}}]^T$, the task completion quality constraints can be represented as

$$\begin{aligned} \sum_{j=1}^{N}\sum_{m=1}^{M} \mathcal{L}_{m,j}\mathcal{W}_{m,k}q_{m,k} \cdot \mathcal{P}_{m,k} \cdot \mathcal{D}_{j,k} &= \boldsymbol{\mathcal{P}}_k^T \boldsymbol{A}_k \boldsymbol{\mathcal{D}}_k \\ &= \begin{bmatrix} \boldsymbol{\mathcal{P}}_1 \\ \boldsymbol{\mathcal{P}}_2 \\ \vdots \\ \boldsymbol{\mathcal{P}}_K \\ \boldsymbol{\mathcal{D}}_1 \\ \boldsymbol{\mathcal{D}}_2 \\ \vdots \\ \boldsymbol{\mathcal{D}}_K \end{bmatrix}^T \begin{array}{cc} & \begin{array}{ccccccc} \phantom{\boldsymbol{O}} & & k\downarrow & & k+K\downarrow & & 2K\downarrow \end{array} \\ \begin{array}{c} \\ \\ k\rightarrow \\ \\ k+K\rightarrow \\ \\ 2K\rightarrow \end{array} & \begin{bmatrix} \boldsymbol{O} & \cdots & \boldsymbol{O} & \cdots & \boldsymbol{O} & \cdots & \boldsymbol{O} \\ \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ \boldsymbol{O} & \cdots & \boldsymbol{O} & \cdots & \frac{1}{2}\boldsymbol{A}_k & \cdots & \boldsymbol{O} \\ \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ \boldsymbol{O} & \cdots & \frac{1}{2}\boldsymbol{A}_k^T & \cdots & \boldsymbol{O} & \cdots & \boldsymbol{O} \\ \vdots & \ddots & \vdots & \ddots & \vdots & \ddots & \vdots \\ \boldsymbol{O} & \cdots & \boldsymbol{O} & \cdots & \boldsymbol{O} & \cdots & \boldsymbol{O} \end{bmatrix} \end{array} \begin{bmatrix} \boldsymbol{\mathcal{P}}_1 \\ \boldsymbol{\mathcal{P}}_2 \\ \vdots \\ \boldsymbol{\mathcal{P}}_K \\ \boldsymbol{\mathcal{D}}_1 \\ \boldsymbol{\mathcal{D}}_2 \\ \vdots \\ \boldsymbol{\mathcal{D}}_K \end{bmatrix} \\ &= \boldsymbol{x}^T \widetilde{\boldsymbol{A}}_k \boldsymbol{x} \end{aligned} \tag{23}$$

where $\boldsymbol{A}_k$ is defined as $\boldsymbol{A}_k \triangleq \boldsymbol{\mathcal{L}} \circ \boldsymbol{\mathcal{W}}_k \circ \boldsymbol{q}_k$ ($\boldsymbol{A} \circ \boldsymbol{B}$ means the Hadamard product of matrices $\boldsymbol{A}$ and $\boldsymbol{B}$). Besides these, the capacity constraints can also be transformed with:

$$\begin{aligned}\sum_{k=1}^{K} \mathcal{D}_{j,k} \leq n_j^{\mathcal{E}}, \forall j &\Longleftrightarrow \sum_{k=1}^{K} \boldsymbol{\mathcal{D}_k} \leq \boldsymbol{n}^{\mathcal{E}} \\ \sum_{k=1}^{K} \mathcal{P}_{m,k} \leq n_m^{\mathcal{U}}, \forall m &\Longleftrightarrow \sum_{k=1}^{K} \boldsymbol{\mathcal{P}_k} \leq \boldsymbol{n}^{\mathcal{U}}\end{aligned} \tag{24}$$

Then we have the simplified form of the CSDATAM$_2$-C:

$$P_C^{'} : \quad \min \begin{bmatrix} \boldsymbol{c} \\ \nu \cdot \mathbb{1} \end{bmatrix}^T \boldsymbol{x} \tag{25}$$

$$s.t. \quad \boldsymbol{x}^T \widetilde{\boldsymbol{A}}_k \boldsymbol{x} \geq \varepsilon_k, \forall k \tag{26}$$

$$\begin{bmatrix} \mathbb{1} & \boldsymbol{O} \\ \boldsymbol{O} & \mathbb{1} \end{bmatrix} \boldsymbol{x} \leq \begin{bmatrix} \boldsymbol{n}^{\mathcal{U}} \\ \boldsymbol{n}^{\mathcal{E}} \end{bmatrix} \tag{27}$$

$$\begin{bmatrix} \boldsymbol{O} & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{diag}(\mathbb{1}) \end{bmatrix} \boldsymbol{x} \leq \mathbb{1} \tag{28}$$

$$\boldsymbol{x} \in \mathbb{N}^{(M+N) \cdot K} \tag{29}$$

and the simplified form of the CSDATAM$_2$-Q problem:

$$P_Q^{'} : \quad \max \sum_{k=1}^{K} \boldsymbol{x}^T \widetilde{\boldsymbol{A}}_k \boldsymbol{x} \tag{30}$$

$$s.t. \quad \boldsymbol{x}^T \widetilde{\boldsymbol{A}}_k \boldsymbol{x} \geq \varepsilon_k, \forall k \tag{31}$$

$$\begin{bmatrix} \boldsymbol{c} \\ \nu \cdot \mathbb{1} \end{bmatrix}^T \boldsymbol{x} \leq C^{\star} \tag{32}$$

$$\begin{bmatrix} \mathbb{1} & \boldsymbol{O} \\ \boldsymbol{O} & \mathbb{1} \end{bmatrix} \boldsymbol{x} \leq \begin{bmatrix} \boldsymbol{n}^{\mathcal{U}} \\ \boldsymbol{n}^{\mathcal{E}} \end{bmatrix} \tag{33}$$

$$\begin{bmatrix} \boldsymbol{O} & \boldsymbol{O} \\ \boldsymbol{O} & \boldsymbol{diag}(\mathbb{1}) \end{bmatrix} \boldsymbol{x} \leq \mathbb{1} \tag{34}$$

$$\boldsymbol{x} \in \mathbb{N}^{(M+N) \cdot K} \tag{35}$$

Obviously, the CSDATAM$_2$M-C and CSDATAM$_2$M-Q problems are quadratic constraint linear programming problems (special cases of the second-order cone programming problem), if ignore the fact that $\boldsymbol{x} \in \mathbb{N}$. Therefore, we can apply the Branch-and-Bound (BnB) framework here to search for the optimal solutions to these problems step by step [11]. And in every step, we will relax the constraint of variable $\boldsymbol{x}$ to $\mathbb{R}$ so that convex solvers like **CPLEX** and **Gurobi** can be applied to solve these classical problems.

5 Experiments and Analysis

Due to the lack of well-adopted platforms and datasets, in this work, we generate our experimental data in a synthetic way. Based on the simulation data, complete experiments are conducted so that the impacts of different factors are explored.

5.1 Baselines and Comparisons

As the problems will be solved with exact solutions after they are translated into the MIQP problem, it is not necessary to compare it with other baselines except the ones that consider the trade-off between accuracy and running time. Thus, we choose some popular and representative mathematical approaches for the programming program problem as baselines. The chosen approaches are:

1. **Genetic algorithm, GA.** Genetic algorithm is one of the famous methods [12] which can be used for this purpose. GA simulates the evolution of populations with operations like selection, crossover, and mutation. It is designed to favor chromosomes with the highest fitness values to produce the next populations (solutions). As a result, the quality of solutions for a problem is gradually improved until the optimal answer is reached.
2. **Tabu search algorithm, TS.** Tabu search algorithm [13] is a meta-heuristic search method employing local search methods used for mathematical optimization, where the local searches take a potential solution to a problem and check its immediate neighbors in the hope of finding an improved solution. Local search methods have a tendency to become stuck in suboptimal regions or on plateaus where many solutions are equally fit.

In Fig. 3, we can find that not all the baseline approaches can find the optimal task assignment and service deployment schemes. Compared to the TS, the GA shows better performance in obtaining the schemes with best quality, but fail to have a good probability to obtain the best cost. However, different from the exact results generated by solving the mixed integer quadratic programming problems, these heuristic algorithms may always sacrifice their accuracy for the time complexity to find the best schemes.

5.2 Impact of System Configurations

To explore how the system configurations will impact the optimal task assignment and service deployment schemes, we conducted a series of parametric experiments in this section.

Impact of User Number: In Fig. 4, we've shown how the average values of total cost and total task completion quality will change with the increasing of user number in problem CSDATAM$_2$-C and CSDATAM$_2$-Q. It can be found that as M increases, the total cost shows a downward trend in CSDATAM$_2$-C while the total task completion quality increase in CSDATAM$_2$-Q. Obviously, both the former decline and the latter rise are mainly due to the potential appropriate users brought by the increase of user number.

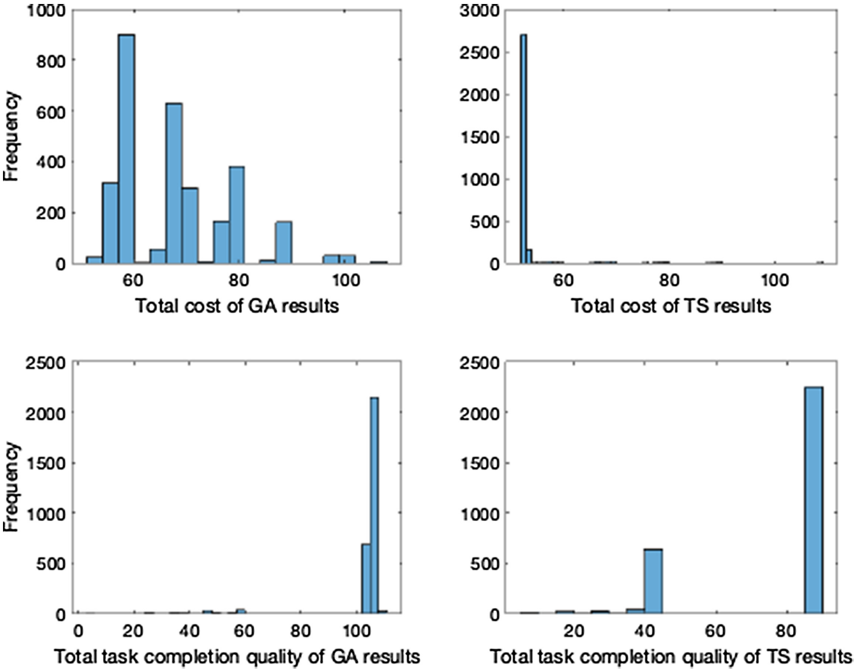

Fig. 3. The comparison with baselines

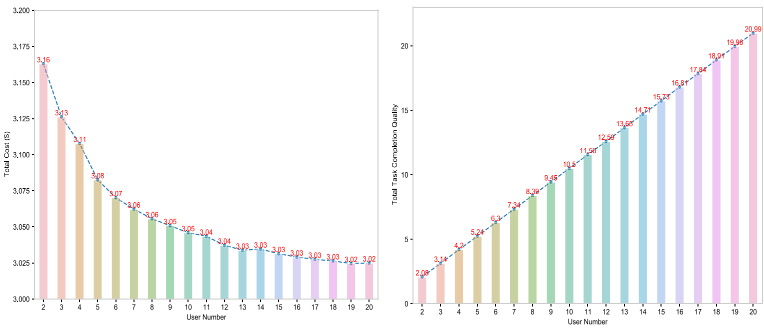

Fig. 4. The impact of user number (left: CSDATAM$_2$-C, right: CSDATAM$_2$-Q)

Impact of Edge Server Number: In Fig. 5, we've shown how the average values of total cost and total task completion quality will change with the increasing of edge server number in problem CSDATAM$_2$-C and CSDATAM$_2$-Q. It can be found that as N increases, the total cost shows a downward trend in CSDATAM$_2$-C while the total task completion quality increases in CSDATAM$_2$-Q. Obviously, both the former decline and the latter rise are mainly due to the potential appropriate edge servers for related services brought by the increase of edge server number. Besides this, we can also find that they become stable

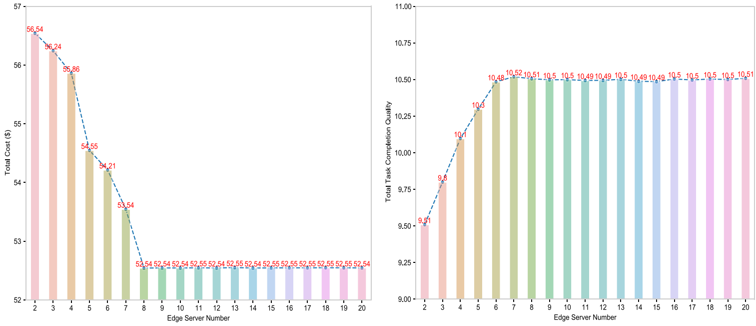

Fig. 5. The impact of edge server number (left: CSDATAM$_2$-C, right: CSDATAM$_2$-Q)

when N is larger than some specific values. This is because we don't need more edge servers if all appropriate users are assigned with matching tasks are related services are deployed near them.

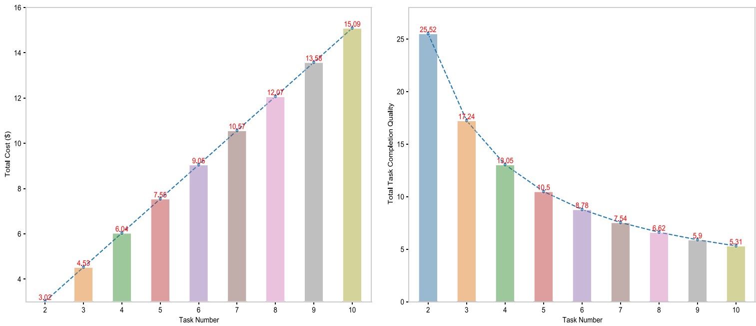

Fig. 6. The impact of sub-task number (left: CSDATAM$_2$-C, right: CSDATAM$_2$-Q)

Impact of Sub-task Number: In Fig. 6, we've shown how the average values of total cost and total task completion quality will change with the increasing of sub-task number in problem CSDATAM$_2$-C and CSDATAM$_2$-Q. It can be found that as K increases, the total cost shows an upward trend in CSDATAM$_2$-C while the total task completion quality decreases in CSDATAM$_2$-Q. The cost rise is due to the increasing number of users to fulfill the task when K increases, so more cost are paid for these additional users. Similarly, the task completion quality decreases because there is no surplus budget to search for better qualities.

Impact of User Willingness: User willingness is ω measured with the percentage of users who would like to be assigned with tasks. In Fig. 7, we've shown how

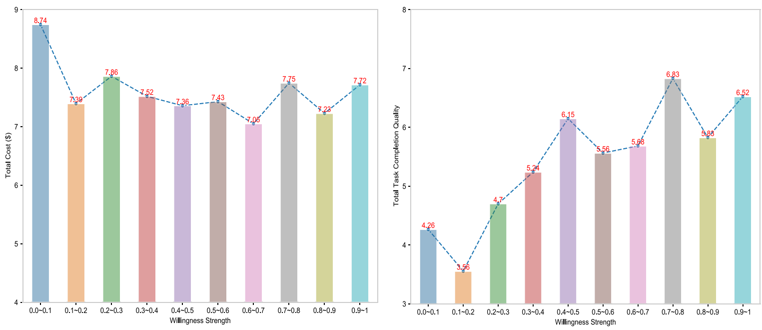

Fig. 7. The impact of user willingness (left: CSDATAM$_2$-C, right: CSDATAM$_2$-Q)

the average values of total cost and total task completion quality will change with the increasing of willing strength ω in problem CSDATAM$_2$-C and CSDATAM$_2$-Q. It can be found that as ω increases, the total cost shows a downward trend in CSDATAM$_2$-C. This decline is mainly due to the external possibilities brought by the increase in ω. However, the task completion quality shows an opposite trend in the result of CSDATAM$_2$-Q, as the external possibilities make it possible for the application developers to select the users with better task completion qualities and lower incentive requirements.

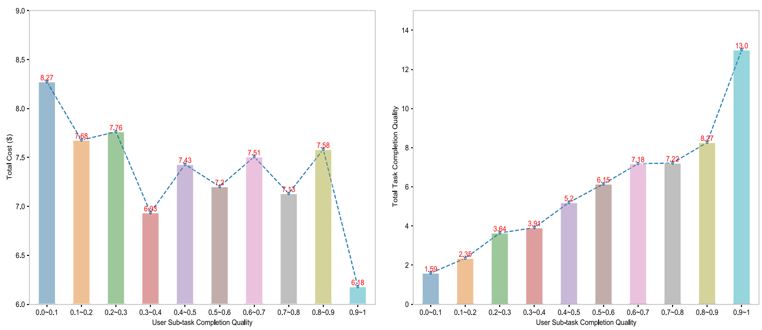

Fig. 8. The impact of users' $\bar{q}$ (left: CSDATAM$_2$-C, right: CSDATAM$_2$-Q)

Impact of User Sub-task Completion Quality: In Fig. 8, we've shown how the average values of total cost and total task completion quality will change with the increasing of users' average task completion quality in problem CSDATAM$_2$-C and CSDATAM$_2$-Q. It can be found that as the user average task completion quality q increases, the total cost shows a downward trend in the result of CSDATAM$_2$-C. This decline is mainly due to the easier satisfaction of the sub-task quality requirements brought by the increase of q. However, the total task completion quality shows an opposite trend in the result of CSDATAM$_2$-Q. This is because the total task completion quality is proportional to the users' task completion qualities.

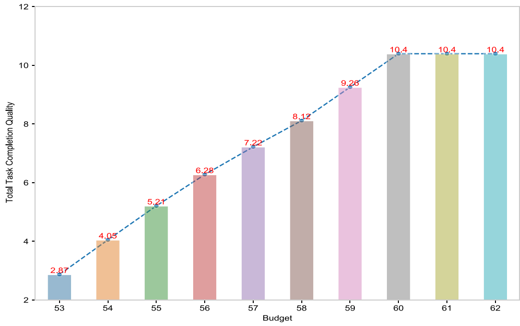

Fig. 9. The impact of cost budget

Impact of Cost Budget: In Fig. 9, we've shown how the average values of total cost will change with the increasing of user average task completion quality in problem CSDATAM$_2$-Q (because it will not effect the results of CSDATAM$_2$-C). It can be found that as the developer's budget on user incentives $C^\star$ increases, the total task completion quality Q shows an upward trend. This increase shows that "the more you pay, the more you get"—the abundant budget make it possible to assign tasks to the users who are competent enough but charge more.

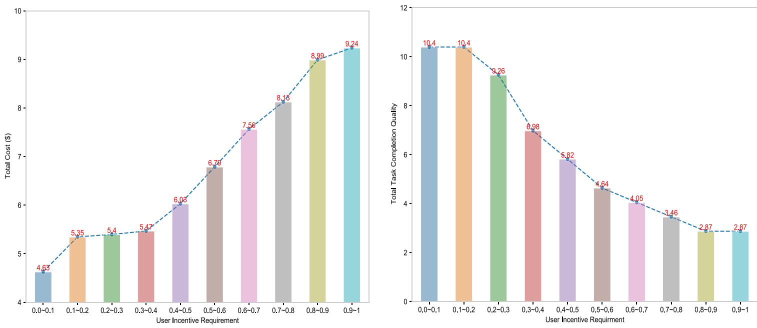

Fig. 10. The impact of users' $\bar{c}$ (left: CSDATAM$_2$-C, right: CSDATAM$_2$-Q)

Impact of User Incentive Requirement: In Fig. 10, we've shown how the average values of total cost and total task completion quality will change with the increasing of user incentives requirement in problem CSDATAM$_2$-C and CSDATAM$_2$-Q. It can be found that as the user incentive requirement c increases, the total cost C_r shows an upward trend in CSDATAM$_2$-C. This is quite easy to understand because the total cost is proportional to the users' incentive requirements. On the other hand, we can also find that the average values of total task completion quality decreases fast with the increasing of c. This is because the limited budget cannot support the pursuit for higher task completion quality—though the optimum can also be found, but the value of it may be smaller than that of a larger space.

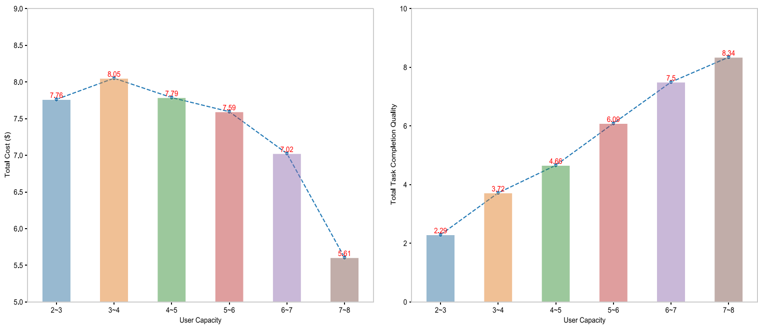

Fig. 11. The impact of user capacity (left: CSDATAM$_2$-C, right: CSDATAM$_2$-Q)

Impact of User Capacity: In Fig. 11, we've shown how the average values of total cost C_r and total task completion quality Q will change with the increasing of user capacity in problem CSDATAM$_2$-C and CSDATAM$_2$-Q. It can be found that as the user capacity n^U increases, the total cost C_r shows a downward trend in CSDATAM$_2$-C. This decline is mainly due to the external possibilities brought by the increase of n^U. This is not difficult to understand because, at first, C_r will not become larger for its optimality. Then if two users have the same task completion quality but different incentive requirement, the sub-task which are originally assigned to the expensive one will be assigned to another. As the reason also works well in the CSDATAM$_2$-Q problem, then we can see that the average values of total task completion quality shows an upward trend.

Impact of Edge Server Capacity: In Fig. 12, we've shown how the average values of total cost and total task completion quality will change with the increasing of edge server capacity in problem CSDATAM$_2$-C and CSDATAM$_2$-Q. It can be found that they do not give exact changing rules with the increasing of the edge server capacity n^U. Actually, this is because here in our model, we've

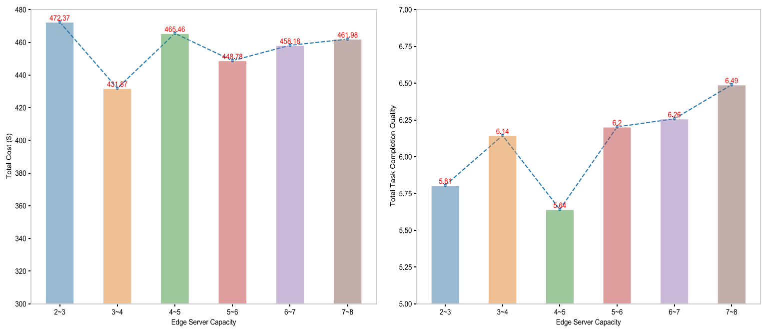

Fig. 12. The impact of edge capacity (left: CSDATAM$_2$-C, right: CSDATAM$_2$-Q)

assumed that the deployment cost on different edge servers are the same so that the objectives will not be effected much when it changes. On the contrary, as the services should be deployed on the edge servers near the users who are assigned with related tasks, the main factor will be the user locations.

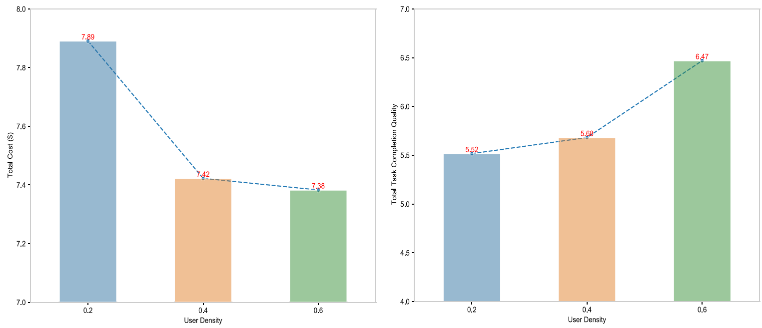

Fig. 13. The impact of user density (left: CSDATAM$_2$-C, right: CSDATAM$_2$-Q)

Impact of User Distribution: To explore the impact of user distribution, here we first use an index called "user density" to measure it. Heuristically, we define it with:

$$\rho_u = 1 - \frac{\min |\{e_{j_i}| \sum_{i=1}^{n} \sum_{m=1}^{M} \mathcal{L}_{m,j_i} \geq 0.8N, e_{j_i} \in \mathcal{E}\}|}{N}. \tag{36}$$

Obviously, the user density will be $1 - 1/10 = 0.9$ if users gather around 1 edge server of 10 edge servers, and will be $1 - 10/10 = 0$ if they are evenly distributed.

In Fig. 13, we've shown how the average values of total cost and total task completion quality will change with the increase of user density in problem

CSDATAM$_2$-C and CSDATAM$_2$-Q. It can be found that as the user density ρ_u increases, the total cost shows a downward trend. This decline is mainly cost in the repeated deployment of services—it will be not necessary for the application developers to deploy the services on several edge servers if the workers of the same sub-tasks are in the same serving area of an edge server. Based on this, we can also see the upward trend of the total task completion quality—as mentioned above, the adequate budget can bring better quality.

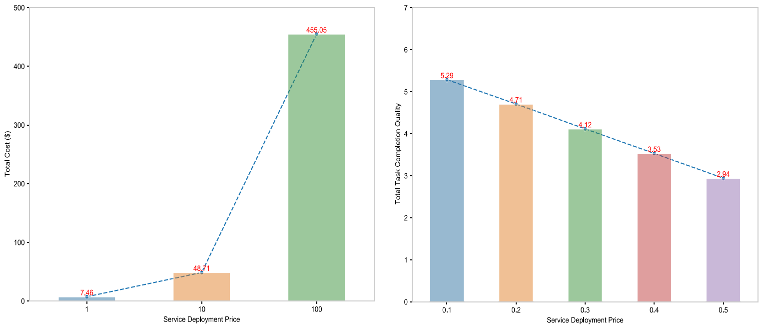

Fig. 14. The impact of deployment price (left: CSDATAM$_2$-C, right: CSDATAM$_2$-Q)

Impact of Deployment Price: In Fig. 14, we've shown how the average values of total cost and total task completion quality will change with the increase of service deployment price ν in problem CSDATAM$_2$-C and CSDATAM$_2$-Q. It can be found that as the service deployment price ν increases, the total cost also increases fast. This rise is mainly due to the fact that the total cost is proportional to ν. Similarly, the developer's budget will strongly limit the total task completion quality with the increasing cost in satisfying the users' requirements, so the decline of it will be obvious.

6 Related Work

In this section, we will review some representative works that are related to our problem to see what efforts are made to solve this kind of problem.

6.1 The Incentive Mechanism in Mobile Crowd Sensing

First, we are to show some research about the incentive mechanism in the MCS environment. Deterding et al. [14] made some of the earliest contributions to the incentive mechanism of group intelligence perception, they introduced economic models into the incentive mechanism and proposed methods of dynamic prices and virtual points to ensure the participation rate and minimize cost.

For users who are rarely selected as participants, there is a possibility of losing interest and withdrawing from the group intelligence perception system. Gao et al. [15] therefore proposed the long-term problem of participation rate, and solved this problem based on the Lyapunov-based auction model. Besides this, Sun et al. introduced the restless multi-armed bandit (MAB) process model and the heterogeneous belief values model [16], they transformed this continuous group intelligence incentive model into a MAB process based on the social status and real-time to solve this long-term incentive problem. The location distribution of the participants affects the quality of the task. The server platform should not only recruit more participants at minimal cost but also consider the location distribution of users. Therefore, Jaimes et al. [17] proposed the GIA algorithm in combination with the Greedy Budgeted Maximum Coverage algorithm to improve the coverage of the area of interest for a given budget.

6.2 The Task Assignment Research

Task assignment is a popular problem in the research about MEC. It is to perform task allocation so that the latency and energy consumption of mobile application execution can be dynamically adjusted to make the task allocation results meet the user requirements. To address the above problem, existing research has proposed several approaches to meet the requirements: Dinh et al. [18] used randomly generated task graphs to allocate some tasks and minimize energy consumption. Kao et al. [19] used integer programming to calculate task allocation and scheduling strategies to save network resources as much as possible. Literature [20] studied to minimize time delay under the constraints of given network resources, and proposed an efficient polynomial-time algorithm to complete task assignment. Sardellitti et al. [21] proposed a task assignment algorithm based on a directed acyclic graph (DAG) model that minimizes the energy consumption of mobile devices under a given time delay constraint. Deng et al. [22] decomposed mobile applications into multiple subtasks, built a DAG model, and modeled the task allocation problem as a nonlinear 0–1 programming problem to achieve the goal of minimizing energy consumption. Kwak et al. [23] designed a dynamic migration algorithm based on Lyapunov's optimization technology to ensure that the system delay is controlled within a given range and to minimize the energy consumption of mobile devices. Jiang and You et al. [24,25] further extended the work of [23] to multi-core mobile devices.

These researches shed light on the fundamental concepts and inspired the thought of related optimizing in mobile crowd sensing systems and multi-access edge computing systems. Based on these works, we try to go further in balancing the performance and cost for the MEC-based MCS system. We will combine the advantages of using incentive mechanisms and the methods of assigning tasks to generate an appropriate strategy that can have a trade-off between performance (quality) and cost.

7 Conclusion and Future Work

In this paper, we first introduce the MEC-based MCS system and describe how the MEC architecture can be used in improving the performance of the mobile crowd sensing system. Then, considering the incentive mechanism and the quality model of complex tasks, we build up a cost-quality analysis framework for the task assignment and service deployment problem in this system. Finally, we solve the problem after translating it into a mixed-integer quadratic programming problem and investigate the factors that may impact the results. In the future, we plan to refine our model and the analysis framework to make it more practical, and we are going to incorporate dynamic factors into this problem to further improve the performance of the proposed approach.

Acknowledgement. This research was partially supported by the National Key Research and Development Program of China (No. 2017YFB1400601), Key Research and Development Project of Natural Science Foundation of China (NO. 61772461, No. 61802343, No. 62072402) and Zhejiang Provincial Natural Science Foundation of China (No. LQ21F020007, No. LQ20F020015, No. LR18F020003).

References

1. Chen, Y., Zhou, M., Zheng, Z., Chen, D.: Time-aware smart object recommendation in social Internet of Things. IEEE Internet Things J. **7**(3), 2014–2027 (2020)
2. Deng, S., Zhao, H., Fang, W., Yin, J., Dustdar, S., Zomaya, A.Y.: Edge intelligence: the confluence of edge computing and artificial intelligence. IEEE Internet Things J. **7**(8), 7457–7469 (2020)
3. Shi, W., Cao, J., Zhang, Q., Li, Y., Xu, L.: Edge computing: vision and challenges. IEEE Internet Things J. **3**(5), 637–646 (2016)
4. Ren, J., Pan, Y., Goscinski, A., Beyah, R.A.: Edge computing for the Internet of Things. IEEE Netw. **32**(1), 6–7 (2018)
5. Wang, D., Xu, D., Yu, D., Xu, G.: Time-aware sequence model for next-item recommendation. Appl. Intell. 1–15 (2020). https://doi.org/10.1007/s10489-020-01820-2
6. Filippini, I., Sciancalepore, V., Devoti, F., Capone, A.: Fast cell discovery in mm-wave 5G networks with context information. IEEE Trans. Mob. Comput. **17**(7), 1538–1552 (2017)
7. Fan, Q., Ansari, N.: Application aware workload allocation for edge computing-based IoT. IEEE Internet Things J. **5**(3), 2146–2153 (2018)
8. Chen, Y., Deng, S., Ma, H., Yin, J.: Deploying data-intensive applications with multiple services components on edge. Mob. Netw. Appl. **25**(2), 426–441 (2020). https://doi.org/10.1007/s11036-019-01245-3
9. Zhao, H., Deng, S., Zhang, C., Du, W., He, Q., Yin, J.: A mobility-aware cross-edge computation offloading framework for partitionable applications. In: Bertino, E., Chang, C.K., Chen, P., Damiani, E., Goul, M., Oyama, K. (eds.) 2019 IEEE International Conference on Web Services, ICWS 2019, Milan, Italy, 8–13 July 2019, pp. 193–200. IEEE (2019)
10. Deng, S., Wu, H., Tan, W., Xiang, Z., Wu, Z.: Mobile service selection for composition: an energy consumption perspective. IEEE Trans. Autom. Sci. Eng. **14**(3), 1478–1490 (2015)

11. Gupta, O.K., Ravindran, A.: Branch and bound experiments in convex nonlinear integer programming. Manag. Sci. **31**(12), 1533–1546 (1985)
12. Deng, S., Huang, L., Taheri, J., Yin, J., Zhou, M., Zomaya, A.Y.: Mobility-aware service composition in mobile communities. IEEE Trans. Syst. Man Cybern. Syst. **47**(3), 555–568 (2017)
13. Liu, H., Zhang, J., Zhang, X., Kurniawan, A., Juhana, T., Ai, B.: Tabu-search-based pilot assignment for cell-free massive MIMO systems. IEEE Trans. Veh. Technol. **69**(2), 2286–2290 (2019)
14. Deterding, S., Dixon, D., Khaled, R., Nacke, L.: From game design elements to gamefulness: defining "gamification". In: Proceedings of the 15th International Academic MindTrek Conference: Envisioning Future Media Environments, pp. 9–15 (2011)
15. Gao, L., Hou, F., Huang, J.: Providing long-term participation incentive in participatory sensing. In: 2015 IEEE Conference on Computer Communications (INFOCOM), pp. 2803–2811. IEEE (2015)
16. Sun, J.: An incentive scheme based on heterogeneous belief values for crowdsensing in mobile social networks. In: 2013 IEEE Global Communications Conference (GLOBECOM), pp. 1717–1722. IEEE (2013)
17. Jaimes, L.G., Vergara-Laurens, I., Labrador, M.A.: A location-based incentive mechanism for participatory sensing systems with budget constraints. In: 2012 IEEE International Conference on Pervasive Computing and Communications, pp. 103–108. IEEE (2012)
18. Dinh, T.Q., Tang, J., La, Q.D., Quek, T.Q.: Adaptive computation scaling and task offloading in mobile edge computing. In: 2017 IEEE Wireless Communications and Networking Conference (WCNC), pp. 1–6. IEEE (2017)
19. Kao, Y.-H., Krishnamachari, B., Ra, M.-R., Bai, F.: Hermes: latency optimal task assignment for resource-constrained mobile computing. IEEE Trans. Mob. Comput. **16**(11), 3056–3069 (2017)
20. Zhang, W., Wen, Y., Wu, D.O.: Collaborative task execution in mobile cloud computing under a stochastic wireless channel. IEEE Trans. Wirel. Commun. **14**(1), 81–93 (2014)
21. Sardellitti, S., Scutari, G., Barbarossa, S.: Joint optimization of radio and computational resources for multicell mobile-edge computing. IEEE Trans. Sig. Inf. Process. Netw. **1**(2), 89–103 (2015)
22. Deng, M., Tian, H., Fan, B.: Fine-granularity based application offloading policy in cloud-enhanced small cell networks. In: 2016 IEEE International Conference on Communications Workshops (ICC), pp. 638–643. IEEE (2016)
23. Kwak, J., Kim, Y., Lee, J., Chong, S.: DREAM: dynamic resource and task allocation for energy minimization in mobile cloud systems. IEEE J. Sel. Areas Commun. **33**(12), 2510–2523 (2015)
24. Jiang, Z., Mao, S.: Energy delay tradeoff in cloud offloading for multi-core mobile devices. IEEE Access **3**, 2306–2316 (2015)
25. You, C., Zeng, Y., Zhang, R., Huang, K.: Asynchronous mobile-edge computation offloading: energy-efficient resource management. IEEE Trans. Wirel. Commun. **17**(11), 7590–7605 (2018)