Hindawi
Applied Computational Intelligence and Soft Computing
Volume 2022, Article ID 2998385, 17 pages
https://doi.org/10.1155/2022/2998385

Research Article

Clustering Ant Colony-Based Edge-Server Location Strategy in Mobile Crowdsensing

Ahmed. A. A. Gad-Elrab and Amin Y. Noaman

Faculty of Computing and Information Technology, King Abdul-Aziz University, Jeddah, Saudi Arabia

Correspondence should be addressed to Ahmed. A. A. Gad-Elrab; aaahmad4@kau.edu.sa

Received 17 June 2022; Revised 5 December 2022; Accepted 9 December 2022; Published 29 December 2022

Academic Editor: Jun He

Recently, edge-based mobile crowdsensing has become an important sensing technology that takes advantage of mobile devices to collect information about surroundings based on using a group of mobile edge servers that are deployed at the network edge as a link between users and the central server for data filtering and aggregation. Each user may collect multiple data types in mobile collective sensing. For facilitating data aggregation, the same data type carried by various users is assumed to be uploaded to the same mobile edge server. The main problem is determining the server which should be activated to process each data type for reducing the overall cost. In this paper, the problem is formulated as one form of the unqualified multicommodity facility location problem. To solve this problem, two edge-server location strategies are proposed, which use a clustering method for dividing the set of mobile users with data items into clusters and use the ant colony approach to select a mobile edge server for each data type in each cluster. Extensive simulations are conducted based on widely used real data sets. The simulation results show that the proposed strategy achieves better performance than the existing methods in terms of service and facility costs.

1. Introduction

In recent decades, a new sensing paradigm appeared, called mobile crowdsensing (MCS), due to the existence of a lot of mobile devices with efficient sensing and powerful capabilities in human life [1–4]. In MCS, mobile devices are used for collecting sensing data form the surrounding environments [5, 6]. This collected data can be used for introducing various services such as construction of radio environment map [7], management of roadside parking [8], and assessment of road surface [9].

Previously, the architecture of traditional MCS is centralized, where there is a central server (CS) that receives directly uploaded sensing data from mobile users. The main drawback of this traditional MCS is that, in the case of large-scale scenarios, the central server may receive a very big amount of data streams from mobile users, which creates a very high load on the CS and networks. In addition, the leak risk of the user privacy increases because all collected data are stored on the CS. Fortunately, due to the faster evolution of Internet-of-Things (IoTs) and 5G communication, the paradigm of mobile edge-computing (MEC) [10–12] is very helpful in solving the problems of the centralized architecture of MCS.

Mobile edge-computing (MEC) [13] can move the computation and processing tasks to mobile edge servers (MESs) that are located near the data source, instead of executing them on the CS [14]. Thus, MCS architecture presents a new layer by distributing set of MESs between the CS and mobile users like a bridge. In this strategy, mobile users can upload sensing data over MESs instead of uploading to CS directly. This layer aggregates and processes uploaded data. Based on data types that are carried by mobile users, the MCS paradigm will guide them to upload data to different edge servers. In other words, each type of data is aggregated on a single MES. Then, the aggregated and processed data are sent to CS for providing the available services of MCS. Aggregating data of the same type on a single MES can filter the redundant sensing data and will remove the erroneous and redundant data. This process

reduces the size of data that will be sent to CS, which decreases the computation load and the network traffic on CS.

In edge-based MCS scenario for collecting data of crowdsensing, the total cost (TC) includes two costs that must be taken into account, which are service cost (SC) and facility cost (FC). The SC represents the cost of movement of the users to upload data, the FC includes the server activation cost (SAC) which represents the cost for activating MES, and the data processing cost (DPC) represents the processing cost of uploaded data. In daily life, usually mobile users stay a long time in a few places such as workplaces and home, and then they tend to leave these places for uploading data and go back to their initiation. Therefore, the SC is the total travelling distance for the user from the initial location, thereafter passing by the corresponding MESs and getting back to the initial location. As shown in Figure 1, the SC for user u_1 is the summation of costs C_1, C_2, C_3, and C_4. The FC for ES S_1 is $C(S_1) + C_1(b_1)$, which includes the costs for SAC and DPC of b_1 type of data, respectively. Thus, based on this scenario, the main problem is which MES should be activated to process each type of data to minimize the total cost (TC). This problem is called mobile edge server activation problem (MESAP).

The edge-based MCS paradigm has been studied within various fields, such as vehicular crowdsensing [16], task allocation [17], and user recruitment [18, 19]. However, none of them takes into account the problem of minimizing the cost of data offloading in the edge-based MCS. Existing works regarding task offloading in MEC concentrate on minimizing the makespan [20, 21] of task execution or overhead [22, 23]. Thus, they do not take into account the movement cost of a user during the data offloading process, and they cannot be applied to the described edge-based MCS scenario directly. In addition, the existing works in MCS did not consider the overhead of data processing which represents the server view, but they focus on minimizing data uploading cost from the user view [24, 25]. The first research that takes into account the server view (facility cost, FC) and the user view (service cost, SC) was presented in [15].

In this paper, to solve the MESAP, the server and user perspectives are considered as presented in [15]. Based on these perspectives, the MESAP will be formulated based on the problem of uncapacitated multicommodity facility location, and two edge-server location strategies are proposed. Each proposed strategy uses a clustering method for dividing the mobile users set with data items into clusters. The first strategy uses the ant colony approach in the first tier to select MES for each data type in a cluster. Then, it merges all the selected sets of MESs and uses a simple heuristic method in the second tier to reallocate each data type to its appropriate mobile edge server, while the second strategy uses the ant colony approach in the two tiers to do the selection and reallocating processes.

The major contributions of this paper are described as follows:

(i) Formulating the MESAP as an uncapacitated multicommodity facility location problem

(ii) Using clustering to divide the mobile users set with data items into clusters

(iii) Using the ant colony approach for selecting the appropriate ES for each data type

(iv) Proposing a new heuristic strategy called one-tier ant colony clustering-based strategy for solving this problem

(v) Proposing a new heuristic strategy called two-tier ant colony clustering-based strategy for solving this problem

(vi) Studying and simulating the performance of the proposed strategies using data sets is used widespread in real world: epfl/mobility, roma/taxi, and geolife trajectory

What remains of the paper is organized in the following manner. The related works are reviewed in Section 2. In this section, the mobile edge server activation problem (MESAP) is described, and also the proposed clustering ant-colony-based strategies. In Section 3, the evaluation and simulation results are conducted to evaluate the proposed strategies performance. Finally, the last section concludes this paper.

2. Related Works

2.1. Edge-Based Mobile Crowdsensing (EMCS). There are some approaches that have been proposed for MCS based on distributed architectures. In [26], a new anonymized data-collection method was proposed for estimating data distributions. In [27], the authors studied the correlation effect of sensing data on differential privacy for protection of MCS systems and introduced two mechanisms of perturbation for two different perspectives. From protector's perspective, they introduced a mechanism that uses the standard definition of differential privacy for deducing the scale value based on the Bayesian network for modelling the probabilistic relevance among sensing data. While in adversary's perspective, they proposed a mechanism that analyzes the importance of maximum correlated group for computing the Bayesian differential privacy leakage based on Gaussian correlation model for describing the data correlation.

In [28], the authors proposed two strategies for managing privacy preserving reputation and handling malicious participants in MCS based on edge computing. Marjanovic et al. [29] proposed MEC paradigm for MCS for increasing the quality of service in MCS. In [16], the authors introduced an edge-based framework in applications of largescale vehicular crowdsensing for minimizing the energy consumption of participating vehicles in the heterogeneous crowdsensing applications. The authors in [30] proposed an edge-based network selection scheme in vehicular crowdsensing and formulated the problem as an optimization problem with double objectives to maximize user satisfaction. In [31], based on edge computing, the authors proposed a distributed ledger framework in MCS for supporting decentralized incentives. In [32], the authors introduced a mechanism of edge-assisted incentive in MCS to satisfy the individual rationality and truthfulness. In [17], the authors proposed a fog-assisted task allocation method in MCS. In addition, a scheme for secure data deduplication with fog-

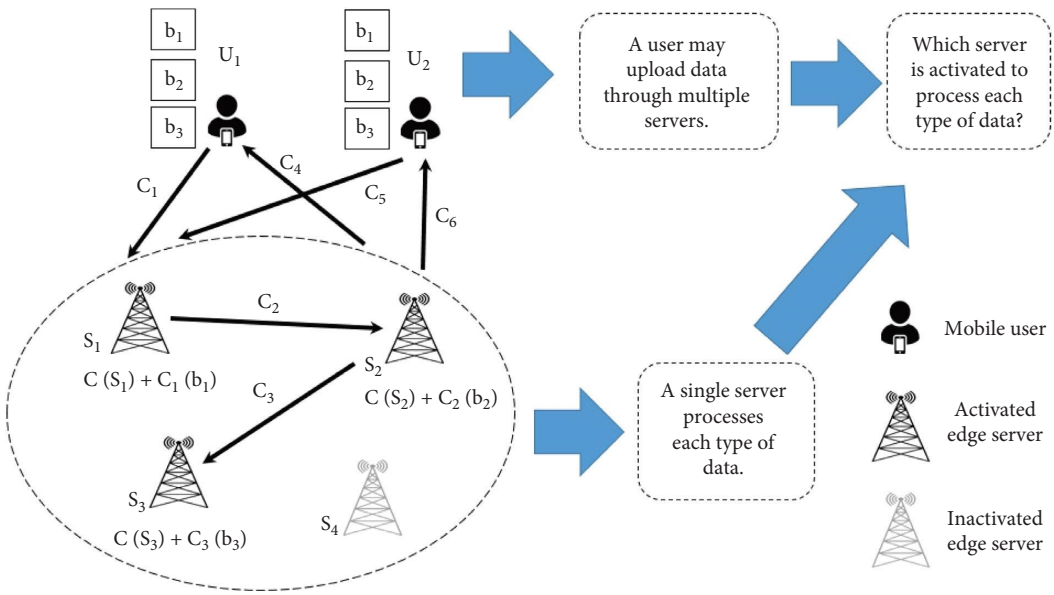

FIGURE 1: Problem description in a system of edge-based MCS for user u_1 who spends movement costs C_1, C_2, C_3, and C_3 for uploading data and for server S_1 which has cost $C(S_1)$ for activating the server and $C_1(b_1)$ to process data b_1 (a modified example based on [15]).

assisted method was proposed for improving the efficiency of communication. Furthermore, there are some approaches that have been proposed by concentrating on the user recruitment in edge-aided MCS [18, 19, 33]. In [18], for sparse data collection, the authors investigated the user recruitment. In [19, 33], the authors proposed a mechanism for incentive-aware recruitment in edge-based MCS. In addition, in [15], the authors studied the edge-server location problem in MCS and proposed a strategy for edge-server location to minimize the crowdsensing cost, which considers the facility cost (server perspective) and the service cost (user perspective). However, the authors in [15] did not consider the load balance among MESs in MCS scenario.

Aforementioned approaches explore the edge-based MCS from different aspects. Nevertheless, none of them takes into account all aspects in terms of facility cost, service cost, and load balance. Therefore, in this paper, new edge-server location strategies are proposed to solve edge-server location problem by considering all of these issues.

2.2. The Problem of Facility Location (FLP). The problem of facility location (FLP) is a classic optimization problem that determines the best location for a warehouse or factory to be placed based on facility costs, transportation distance, and geographical demand. FLP aims to maximize the profit of a supplier based on the given location and demand of the customer. FLP has stirred the interest of numerous researchers. Based on the capacity of the facility, FLP can be categorized into the capacitated facility location [34–36] and uncapacitated facility location [37–40]. In addition, there are numerous shapes of the classical FLP. For example, the problem of k-median is a kind of FLP which has a limitation on the number of opened facilities. In [41], the authors formulated the problem of total movement minimization of clients and facilities as a problem of k-median. To solve the problem of a k-median, the authors in [42] proposed an algorithm of greedy local search. The k-level uncapacitated FLP is another shape of FLP where the demands must be moved between the facilities in a hierarchical order. For the 2-level FLP, there are some works, such as [37, 38], which proposed the approximation algorithm. In addition, [43] introduced a logarithmic approximation algorithm for the multilevel FLP. On the other hand, a client can request a subset of commodities, and in this case, the FLP problem is called multicommodity facility location (MFL). In [39, 44], for uncapacitated MF, the authors proposed approximation algorithms, while the authors in [35] proposed a wide-ranging approach for capacitated MFL. Furthermore, some works take the facility disruption into account, where some failed facilities may be subjected [45–47].

In this paper, FLP will be represented based on the problem of uncapacitated multicommodity facility location as it was considered in [15]. This form of FLP is different from the classical MFL, which takes into consideration two constraints which are as follows: (1) each commodity can be served only by single facility and (2) the travelling distance between various facilities. These two constraints make FLP problem harder than the classical MFL, so most of the present approaches are not applicable directly for solving the FLP problem.

Based on the above-mentioned ideas, the main challenges are as follows: (1) the simplest FLP is NP-hard, (2) there are various data types, and the distance of travelling among mobile edge servers is considered to determine a facility location strategy with minimum cost, so it is more difficult than the traditional FLP, where there is only one kind of data, and (3) the traditional combination optimization approaches could not work well in case of multiple data types scenario.

2.3. Mobile Edge Server Activation Problem (MESAP)

2.3.1. Assumptions and Models. Here, the MCS scenario is described as follows: (1) firstly, moving users collect sensing data, (2) moving users upload data to the mobile edge servers, (3) mobile edge servers execute data filtering and aggregation, (4) aggregated data are sent to the central cloud server, and (5) finally, the central cloud server analyzes the data and then it generates the knowledge that will be used for providing the services of MCS.

To satisfy the previous mentioned scenario, an architecture edge-based MCS is required. Here, this architecture contains a cloud central server, CS, a set of mobile edge servers, $\text{MES} = \{s_1, s_2, \ldots, s_j, \ldots, s_M\}$, where M is the total number of mobile edge servers, and a set of mobile users $MU = \{u_1, u_2, \ldots, u_i, \ldots, u_N\}$, where N is the total number of mobile users. In addition, there is a data type set denoted as $B = \{b_1, b_2, \ldots, b_t, \ldots, b_K\}$, where K is the total number of data types. Each mobile edge server s_j is able to operate in any configuration $\beta_j \in 2^B$ and the cost of processing of this combination of data types is denoted as $PC(\beta_j)$. Each mobile edge server s_j has a cost of activation AC_j and for each data type b_t, there is an incremental cost of processing $IPC_j(b_t)$. Therefore, the facility cost FC_j to activate edge server s_j with configuration β_j is defined as follows:

$$FC_j = AC_j + \sum_{b_t \in \beta_j} IPC_j(b_t). \tag{1}$$

Each user u_i in MU has a set of data items denoted as $D_i = \{d_1^i, d_2^i, \ldots, d_x^i, \ldots, d_{X_i}^i\}$, where each data item d_x^i has a data type b_t in B. So, each user can carry multiple data types denoted as $BT_i \subseteq B$. Here, the service cost SC_i for a mobile user u_i represents the travelling distance of a mobile user in the uploading data process, where any user u_i initiates with an initial location L_i and then moves to its correspondent mobile edge server $MS_i = \{ms_1^i, ms_2^i, \ldots, ms_y^i, \ldots, ms_{Y_i}^i\}$ one after another and finally returns to the initial position. This service cost SC_i is defined as follows:

$$SC_i = DC\left(l_i, ms_1^i\right) + DC\left(ms_{Y_i}^i, l_i\right) + \sum_{ms_y^i \left(\in MS_i / ms_{Y_i}^i\right)} DC\left(ms_y^i, ms_{y+1}^i\right), \tag{2}$$

where $DC(*, *)$ represents the travelling distance cost between two different locations.

Assume the full set of all data items with all users denoted as FD is described as follows:

$$FD = \bigcup_{u_i \in MU} D_i. \tag{3}$$

As shown in Figure 2, for user u_1 that will move towards servers s_1, s_2, and s_3 for uploading data, u_1 will consume the cost $C(u_1, s_1) + C(s_1, s_2) + C(s_2, s_3) + C(s_3, u_1)$, which represents the total distance u_1 travels. While for user u_2 that will go to servers s_1 and s_2 to upload data, u_2 will expend cost $C(u_2, s_1) + C(s_1, s_2) + C(s_2, u_2)$, which represents the total distance u_2 travels. Specifically, the travelling cost between initiation and server is named as $u - s$ service cost, for example, $C(u_1, s_1) + C(s_3, u_1)$ or $C(u_2, s_1) + C(s_2, u_2)$, and the cost for travelling between servers is named as $s - s$ service cost such as $C(s_1, s_2)$. The main used notations in this paper are shown in Table 1.

2.3.2. Problem Formulation. To find a solution for determining which mobile edge servers will be activated and which data types are assigned to the activated mobile edge servers to minimize the total cost, the mobile edge server activation problem (MESAP) will be formulated.

Based on the description of SC_i for each mobile user u_i and FC_j for each mobile edge server s_j, a new variable p_j is proposed to indicate whether the mobile edge server s_j is activated or nonactivated. When s_j is activated, p_j will be 1 and 0 and vice-versa. In addition, the variable $q_j^{b_t} = 1$ is proposed to indicate that the mobile edge server s_j processes b_t type data, and it is 0, otherwise. Variable $w_{ij}^{b_t}$ is 1 if user u_i with b_t data type is assigned to server s_j to upload data. Taking into account that when $b_t = 0, C_j(b_t)$ points to the cost for activating server s_j, consequently, the mobile edge server activation problem (MESAP) is formulated as follows:

$$\textbf{Minimize} \sum_{j=1}^{M} \sum_{t=1}^{K} FC_j * q_j^{b_t} + \sum_{i=1}^{N} SC_i. \tag{4}$$

We have

$$\begin{aligned} &\sum_{j=1}^{M} w_{ij}^{b_t} = 1, \quad \forall b_t \in BT_i, \forall i \in MU, \\ &\sum_{j=1}^{M} q_j^{b_t} = 1, \quad \forall b_t \in B \\ &w_{ij}^{b_t} \le q_j^{b_t} \, \forall b_t \in B, \forall s_j \in MES, \forall i \in MU, \\ &q_j^{b_t} \le p_j \, \forall b_t \in B, \forall s_j \in MES, \\ &p_j, q_j^{b_t}, w_{ij}^{b_t} \in \{0, 1\}. \end{aligned} \tag{5}$$

The first constraint (15) means that there exists MES for processing each type of data carried by each user. The second constraint (16) means that each data type is processed by a single MES. The third constraint (17) means that only when MES has the ability to process the corresponding data type, the user can upload data. The fourth constraint (18) ensures that a mobile edge server has the ability to process data just when it is activated. The fifth constraint (19) shows that the values of decision variables p_j, $q_j^{b_t}$, and $w_{ij}^{b_t}$ are 0 or 1 only. The aim of the proposed strategies is to find the best set of MESs that reduce the total cost and satisfy the above-mentioned constraints.

2.4. The Proposed Ant Colony Clustering-Based Strategies.

In this section, to solve the MESAP, two edge-server location strategies are proposed. The first strategy is called one-tier ant colony clustering-based strategy (OTACS) and the second strategy is called two-tier ant colony clustering-based strategy (TTACS). In the rest of this section, the key idea of the proposed strategies will be introduced, then the two proposed strategies are described in detail.

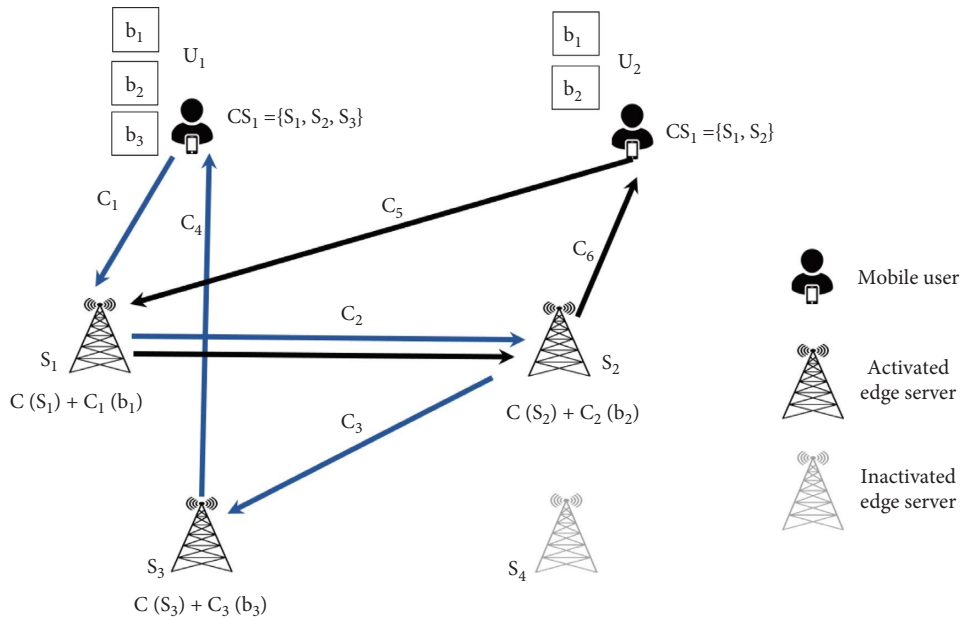

Figure 2: Edge-based MCS system model example. Servers S_1, S_2, and S_3 are chosen to process data types b_1, b_2, and b_3, respectively (a modified example based on [15]).

Table 1: Main used notations.

Symbol	Meaning
CS	The central cloud server
MES	The set of mobile edge servers
MU	The set of mobile users
B	The set of data types
D_i	The set of data items with a user $u_i \in MU$
β_j	The configuration of data types that a mobile edge server $s_j \in MES$ can operate on them
BT_i	The set of data types with a user $u_i \in MU$
MS_i	The corresponding set of mobile edge servers for a user $u_i \in MU$
$PC(\beta_j)$	The processing cost of any configuration β_j (unit: in terms of time, storage, and energy)
$IPC(\beta_j)$	The incremental processing cost of a data type b_t (unit: in terms of time, storage, and energy)
AC_j	The activation cost of mobile edge server (unit: in terms of time, storage, and energy) $s_j \in MES$
FC_j	The facility cost for activating edge server s_j with configuration β_j (unit: in terms of time, storage, and energy)
SC_i	The service cost of a user $u_i \in MU$ (unit: in terms of time, storage, and energy)

2.4.1. Basic Idea. The basic idea of the proposed strategies depends on those as follows: (1) using clustering to divide the set of mobile users with data items into clusters, (2) using the ant colony approach for selecting the appropriate MES for each data type in each cluster, (3) merging the selected subsets of mobile edge servers for all clusters, and (4) reallocating mobile edge server for each type of data for all users based on the merged mobile edge servers set.

To satisfy the basic idea of the proposed strategies, firstly, an overview of ant colony approach will be introduced and then the two proposed strategies OTACS and TTACS will be described in detail.

2.4.2. Overview of the Ant Colony Approach. Ant colony optimization (ACO) is a population-based meta-heuristic technique which depends on the foraging behavior of real ants. These ants forage for food and construct the shortest routes from their nest to the food source. ACO is a class of algorithms which construct their solutions based on the data problem, and it has been presented for application to problems of discrete optimization. In a real environment, ants look for food sources randomly. When an ant discovers a food source, they carry some food back to their colony. Moreover, when they move along the path, they leave a chemical substance known as pheromone while they are

moving. In turn, the higher rate of pheromone trails represents shorter paths. By using pheromone trails as a communication mechanism, each ant makes decisions. The intensity of the pheromone trails left on the ground depends on the quality of the solution (food source) found. Pheromone trails accumulate with multiple ants in shorter paths, resulting in a higher density than in longer paths, therefore increasing its attractiveness. By using an evaporation rate, all pheromone remains are eventually reduced. On the other side, an evaporation process introduces the exploration and prevents staying in a local minimum. At the end of each iteration, the values of pheromone are updated [48, 49]:

$$P_{ij}^{k} = \begin{cases} \dfrac{\left(a_{ij}\right)^{\alpha}\left(b_{ij}\right)^{\beta}}{\sum_{m \in N_i^k}\left(a_{im}\right)^{\alpha}\left(b_{im}\right)^{\beta}}, & j \in N_i^k, \\ 0, & \text{otherwise}, \end{cases} \tag{6}$$

where P_{ij}^{k} is the probability of moving decision of ant k from node i to node j. Such decision depends on the level of pheromone and heuristic information. N_i^k is the set of possible neighborhoods that have not been visited yet by ant k, b_{ij} is a heuristic function, and a_{ij} is the pheromone amount on edge i and j. α and β are the parameters that determine the relative significance of heuristic information and pheromone concentration. The pheromone update can be formulated in the following manner:

$$a_{ij} \leftarrow a_{ij} + \Delta a_{ij}^{k},$$

$$\Delta a_{ij}^{k} = \begin{cases} \dfrac{Q}{f\left(\psi^{k}\right)}, & l_{ij} \in \psi^{k}, \\ 0, & \text{otherwise}. \end{cases} \tag{7}$$

The evaporation update process is given by

$$a_{ij} \leftarrow (1-\rho)a_{ij}, \tag{8}$$

where ρ is the constant reduction factor of all pheromones, $f(\psi^k)$ is the cost of the solution done by ant k, and Q is a constant. The aforementioned optimization process is stopped after a certain amount of iteration.

2.4.3. One-Tier Ant Colony Clustering-Based Strategy (OTACS). The first proposed strategy is called one-tier ant colony clustering-based strategy (OTACS). OTACS consists of four phases as follows:

Phase 1: clustering phase

In this phase, to reduce the candidate number of MESs for selection process, OTACS uses the clustering approach (any clustering approach of existing algorithms for clustering can be used) to divide the set of mobile users MU with their set of data items (FD) based on the initial location of each mobile user into clusters such that each cluster must have at least one mobile edge server. Let us denote a set of clusters as $CL = \{cl_1, cl_2, \ldots, cl_z, \ldots, cl_Z\}$, where Z is the total number of created clusters. Each cluster cl_z has the following sets:

(i) A set of MESs in a cluster, $MS(cl_z) = \{ms_1, ms_2, \ldots, ms_v, \ldots, ms_{V_z}\}$, where V_z is the number of mobile edge servers in this cluster and $MS(cl_z) \subseteq \text{MES}$

(ii) A set of mobile users in a cluster, $UC(cl_z) = \{uc_1, uc_2, \ldots, uc_r, \ldots, uc_{R_z}\}$, where R_z is the number of mobile users in this cluster and $UC(cl_z) \subseteq MU$

(iii) A set of data items for all mobile users in a cluster $DI(cl_z) = \{D_{uc_1}, D_{uc_2}, \ldots, D_{uc_r}, \ldots, D_{uc_{R_z}}\}$, where $\cup_{uc_r \in UC(cl_z)} D_{uc_r} \subseteq FD$

(iv) A set of data types in a cluster, $DT(cl_z) = \{dt_1, dt_2, \ldots, dt_a, \ldots, dt_{A_z}\}$ where A_z is the number of data types in this cluster and $DT(cl_z) \subseteq B$

Here, the K-means algorithm is used to create the required clusters and the optimal number of clusters is obtained by using silhouette method. Assume that there a set of data points $PI = \{pi_1, pi_2, \ldots, pi_l, \ldots, pi_L\}$, then the silhouette coefficient of a point pi_l, $S(pi_l)$, is a measure for cluster validity, which is defined as follows:

$$S(pi_l) = \frac{(b(pi_l) - a(pi_l))}{\max\left(b(pi_l), a(pi_l)\right)}, \tag{9}$$

where $a(pi_l)$ stands for the average distance of that point with all other points in the same cluster, $b(pi_l)$ represents the average distance of that point with all points in the closest cluster to its cluster. The value of $S(pi_l)$ ranges from -1 to $+1$. If the value of $S(pi_l)$ is nearer to 1, then the point is placed in the correct cluster. If the value of $S(pi_l)$ is negative, the point is placed in the wrong cluster. If it is around 0, the object is between the clusters [50].

Then, the average silhouette value average $S(k)$ calculated for all points in all clusters (assume that the number of clusters is k and the number of total data points is L) is as follows:

$$\text{average}\, S(k) = \frac{1}{L}\sum_{l=1}^{L} S(pi_l). \tag{10}$$

If the value of average $S(k)$ is considerably high, then the number of clusters k is optimal; in other words, the clustering structure with k clusters is appropriate. On the other hand, if the value of average $S(k)$ tends to be very less or negative, then the cluster structure with k clusters is not proper, and it may be having either more or lesser number of clusters than the optimal value.

As illustrated in Figure 3, Cluster C_1 has one MES, $\{S_1\}$, three users, $\{u_1, u_2, u_6\}$, and four data types, $\{b_1, b_2, b_3, b_5\}$. Cluster C_2 has one MES $\{S_4\}$, two users $\{u_3, u_7\}$, and three data types $\{b_1, b_2, b_3\}$. Cluster C_3 has

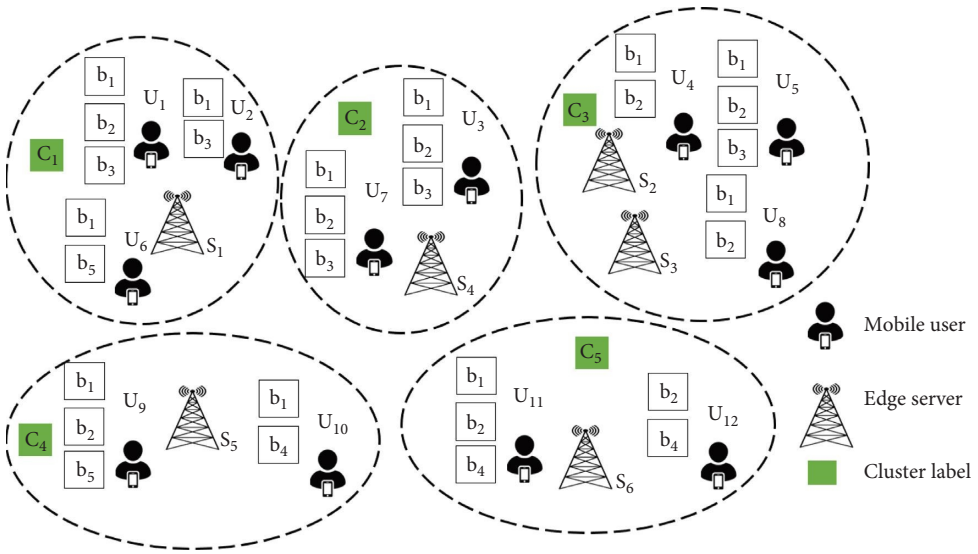

Figure 3: Clustering example for the system model of edge-based MCS.

two MESs, $\{S_2, S_3\}$, three users $\{u_4, u_5, u_8\}$, and three data types $\{b_1, b_2, b_3\}$. Cluster C_4 has one MES, $\{S_5\}$, two users $\{u_9, u_{10}\}$, and four data types $\{b_1, b_2, b_4, b_5\}$. Cluster C_5 has one MES, $\{S_6\}$, two users $\{u_{11}, u_{12}\}$, and three data types $\{b_1, b_2, b_4\}$.

Phase 2: in-cluster allocating phase

In this phase, for each cluster, OTACS uses the ant colony (ACO) approach to select a mobile edge server for each type of data based on the facility cost FC_{ms_v} of each mobile edge server $ms_v \in MS(cl_z)$ and the service cost SC_{uc_r} of each mobile user $uc_r \in UC(cl_z)$. The objective of this phase is defined as follows:

$$\textbf{Minimize} \sum_{v=1}^{V_z} \sum_{a=1}^{A_z} FC_{ms_v} * q_{ms_v}^{dt_a} + \sum_{r=1}^{R_z} SC_{uc_r}. \tag{11}$$

We have

$$\begin{aligned}
&\sum_{v=1}^{V_z} w_{rv}^{dt_a} = 1, \quad \forall dt_a \in DT(cl_z), \forall uc_r \in UC(cl_z),\\
&\sum_{v=1}^{V_z} q_v^{dt_a} = 1, \quad \forall dt_a \in DT(cl_z)\\
&w_{rv}^{dt_a} \le q_v^{dt_a} \forall dt_a \in DT(cl_z), \forall ms_v \in MS(cl_z), \forall uc_r \in UC(cl_z),\\
&q_v^{dt_a} \le p_v \forall dt_a \in DT(cl_z), \forall ms_v \in MS(cl_z),\\
&p_v, q_v^{dt_a}, w_{rv}^{dt_a} \in \{0, 1\}.
\end{aligned} \tag{12}$$

The output of this phase is a set of selected mobile edge servers, SMS_{cl_z}, that represents the most appropriate MESs for filtering and aggregating all data items in this cluster such that each data type is assigned to only one mobile edge server in SMS_{cl_z}.

Phase 3: merging phase

In this phase, OTACS merges the selected sets of mobile edge servers for all clusters in CL into one whole set of selected mobile edge servers, $WS_{CL} \subseteq \text{MES}$, which is defined as follows:

$$WS_{CL} = \bigcup_{z=1}^{Z} \text{SMS}_{cl_z}. \tag{13}$$

Phase 4: reallocating phase

In this phase, OTACS reallocates the mobile edge server for each data type for all users. For each data type b_t, by using the whole set of selected mobile edge servers, WS_{CL}. Assume that the whole set of selected mobile edge servers $WS_{CL} = \{s_1, s_2, \ldots, s_y, \ldots, s_Y\}$, where Y denotes the number of mobile edge servers in WS_{CL}. The objective of this phase is formulated as follows:

$$\textbf{Minimize} \sum_{y=1}^{Y} \sum_{t=1}^{K} FC_y * q_y^{b_t} + \sum_{i=1}^{N} SC_i. \tag{14}$$

We have

$$\sum_{y=1}^{Y} w_{iy}^{b_t} = 1, \quad \forall b_t \in BT_i, \forall i \in MU, \tag{15}$$

$$\sum_{y=1}^{Y} q_y^{b_t} = 1, \quad \forall b_t \in B, \tag{16}$$

$$w_{iy}^{b_t} \le q_y^{b_t} \quad \forall b_t \in B, \forall s_y \in WS_{CL}, \forall i \in MU, \tag{17}$$

$$q_y^{b_t} \le p_y \quad \forall b_t \in B, \forall s_y \in WS_{CL}, \tag{18}$$

$$p_y, q_y^{b_t}, \quad w_{iy}^{b_t} \in \{0, 1\}. \tag{19}$$

Based on this objective, OTACS uses a simple heuristic approach to select the most appropriate mobile edge server for each type of data. OTACS calculates the overall cost for all mobile edge servers in the whole set of selected mobile edge servers, WS_{CL}, and then for each data type b_t, OTACS selects the mobile edge server with the minimum overall cost, $ss_{b_t} \in WS_{CL}$, such that

$$C\left(ss_{b_t}, b_t\right) = \min_{s \in WS_{CL}} \{TC(s, b_t)\}, \tag{20}$$

where $TC(s, b_t)$ represents the overall cost to assign data type b_t to the mobile edge server $s \in WS_{CL}$.

2.4.4. Two-Tier Ant Colony Clustering-Based Strategy (TTACS). In the first proposed strategy, OTACS, the load balancing for mobile edge servers in relation to the number of assigned users and assigned data items for each activated mobile server is not taken into account. Therefore, the second proposed strategy called two-tier ant colony clustering-based strategy (TTACS) is used to tackle this issue. TTACS consists of four phases as the first proposed strategy OTACS, but it has the same first three phases as described OTACS: clustering phase, in-cluster allocating phase, and merging phase. While in the fourth phase, reallocating phase, instead of a simple heuristic which is used in OTACS, TTACS uses the ant colony (ACO) approach to select the most appropriate mobile edge server for each type of data. For each data type, TTACS uses a fitness function which depends on the overall cost for all mobile edge servers in the whole set of selected mobile edge servers, WS_{CL}, and then it constructs the best mobile edge servers set for all data types to be activated in the edge-based scenario of MCS taking into account improving the load balancing on the this activated servers.

Based on these four phases of OTACS and TTACS, they can select the most appropriate mobile edge server for each type of data to be activated in the edge-based MCS scenario. Figure 4 introduces an example of these four phases of the proposed strategies OTACS and TTACS.

2.4.5. Computational Complexity of OTACS and TTACS. Here, the computational complexity of the suggested strategies is going to be described based on the phases of strategy. As shown on the previous two sections, OTACS and TTACS consist of four phases: clustering phase, in-cluster allocating phase, merging phase, and reallocating phase. So, the computational complexity of OTACS and TTACS depends on the complexity of each phase of these four phases which will be described as follows:

(i) *Complexity of Clustering Phase.* In this phase, OTACS and TTACS use the k-means algorithm for creating k clusters, so the computational complexity of this phase is $O(I_1 k(N + M))$, where I_1 is the maximum iterations number for k-means and N and M are the number of mobile users and mobile edge servers, respectively.

(ii) *Complexity of In-Cluster Allocating Phase.* In this phase, OTACS and TTACS use the ACO approach for finding the most appropriate mobile edge servers set in each cluster, so the computational complexity of this phase is $O(I_2 m_c^2 q_c)$, where I_2 is the maximum number of iterations for ACO and m_c and q_c are the number of mobile edge servers and data types in cluster c, respectively.

(iii) *Complexity of Merging Phase.* In this phase, OTACS and TTACS merge all selected sets of mobile edge servers for all clusters, so the computational complexity of this phase is $O(P)$, where $P = |WS_{CL} \le M|$ is the total number of selected mobile edge servers in the whole set.

(iv) *Complexity of Reallocating Phase.* In this phase, OTACS uses a heuristic algorithm for reallocating the set of selected servers, so the computational complexity of OTACS in this phase is $O(KP)$, where K is the total number data types in the system and $P = |WS_{CL} \le M|$ is the total number of selected mobile edge servers in the whole set, While TTACS uses a ACO approach for reallocating the set of selected servers, so the computational complexity of TTACS in this phase is $O(I_3 P^2 M)$, where I_3 is the maximum number of iterations for ACO and $P = |WS_{CL} \le M|$ is the total number of selected mobile edge servers in the whole set.

As a result, the final computational complexity of OTACS and TTACS are $O(I_1 k(N + M)) + O(I_2 m_c^2 q_c) + O(KP)$, and $O(I_1 k(N + M)) + O(I_2 m_c^2 q_c) + O(I_3 P^2 M)$, respectively. Table 2 summaries the final computational complexity of OTACS and TTACS.

3. Evaluation and Simulation Results

3.1. Simulation Settings and Data Preparations. To evaluate the proposed strategies, the performance of six strategies which are low cost first (LF), minimum average distance (DIS), biogeography-based optimization with particle swarm optimization (BBO_PSO) [47], APX2 [15], CMSA [15], and random are compared with the proposed strategies OTACS and TTACS. In LF strategy, a mobile edge server s_j with the lowest processing cost, $IPC_j(b_t)$, is selected for each data type b_t. In DIS, a mobile edge server with the lowest average distance to the mobile users set is selected for each

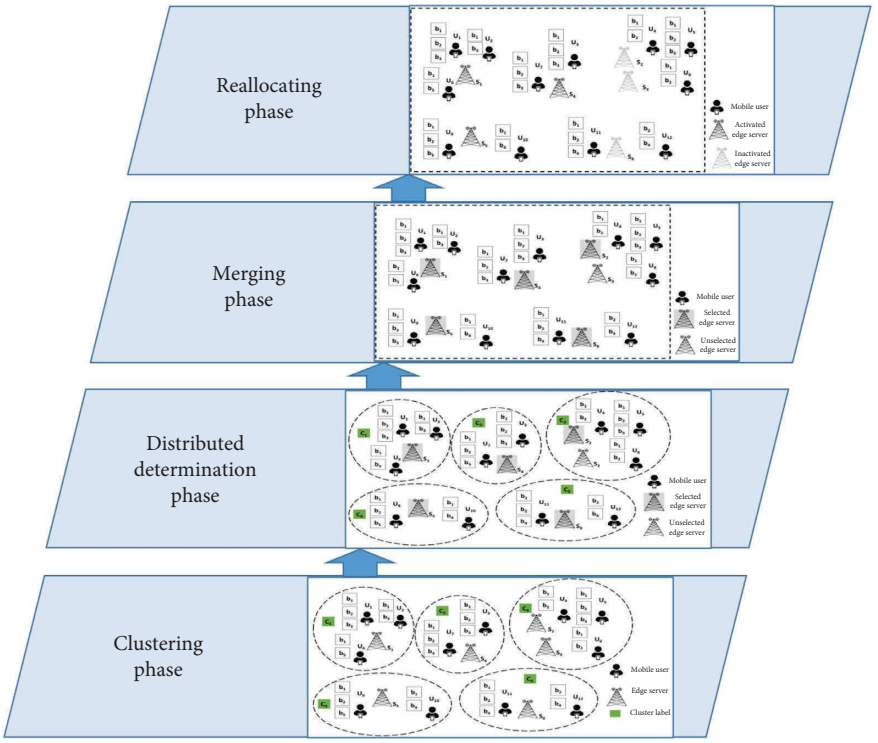

Figure 4: Example of the four phases of the proposed strategies.

Table 2: Summarization of the final computational complexity of OTACS and TTACS.

Phases	OTACS	TTACS	Parameter description
Clustering	$O(I_1k(N+M))$	$O(I_1k(N+M))$	(i) I_1, maximum number of iterations for k-means (ii) N, number of mobile users (iii) M, number of mobile edge servers. (iv) k, number of clusters
In − cluster allocating	$O(I_2m_c^2q_c)$	$O(I_2m_c^2q_c)$	(i) I_2, maximum number of iterations for ACO (ii) m_c, number of mobile edge servers in cluster c (iii) q_c, number of data types in cluster c
Merging	$O(P)$	$O(P)$	(i) $P = \|WS_{CL} \le M\|$, total number of selected mobile edge servers in the whole set
Reallocating	$O(KP)$	$O(I_3P^2M)$	(i) I_3, maximum number of iterations for ACO (ii) K, number data types in the system (iii) $P = \|WS_{CL} \le M\|$, total number of selected mobile edge servers in the whole set
Final complexity	$O(I_1k(N+M)) + O(I_2m_c^2q_c) + O(KP)$	$O(I_1k(N+M)) + O(I_2m_c^2q_c) + O(I_3P^2M)$	

data type. In BBO_PSO, a heuristic algorithm is used to select a mobile edge server which maximizes the total cost fitness for each data type. In random strategy, a mobile edge server is selected randomly for each type of data. In APX2 [15], a mobile edge server is selected by using a relaxation-based approximation approach based facility and service

costs. In CMSA [15], a mobile edge server is selected by using a connected multiagent simulated annealing algorithm based on facility and service costs.

In addition, to evaluate the performance of the suggested strategies to minimize the sensing cost, three data sets are widespread used in real world, which are the dataset of roma/taxi [51], dataset of epfl/mobility [52], and dataset of geolife trajectory [53]. The GPS trajectories in San Francisco Bay Area of approximately 500 taxis, USA, were recorded in the epfl/mobility set over 30 days. The dataset of roma/taxi in in Rome, Italy, that includes approximately 320 taxis' GPS for coordinating mobility traces, collected over 30 days. The dataset of geolife trajectory for in Geolife project by 182 users that includes 17,621 GPS trajectories with a distance of 1.2 million kilometers which were collected.

In the simulation, a uniform distribution is used for generating the facility costs randomly, and the service costs for the mobile users are put as the distance of travel when uploading sensing data in the dataset of real world. Note that the first GPS position of the trajectory of a user is selected as the initial position and the POI positions are selected as a candidate mobile edge server positions.

3.2. Evaluation Metrics. Here, the evaluation metrics which are used to evaluate all strategies are described as follows:

(i) Number of activated mobile edge servers (NAMESs) is the number of selected servers to be activated for processing all data types

(ii) Total cost (TC) is the summation of the facility cost, FC_j, and service cost, SC_i, for all activated MESs and mobile users.

(iii) Load balancing for server-user (LBSU) is the average load balance among the activated MESs based on the number of served mobile users by each activated mobile edge server. Assume that the set of activated MESs is denoted as NAS and the load balance for each activated MES s_j in terms of users is denoted as LBSU_j. The average load balance for server-user, avg LBSU, is defined as follows:

$$\text{avg LBSU} = \frac{\text{Max LBSU} - \text{Min LBSU}}{|\text{NAS}|}, \tag{21}$$

where

$$\begin{aligned} \text{Max LBSU} &= \max_{s_j \in \text{NAS}} \left\{\text{LBSU}_j\right\}, \\ \text{Min LBSU} &= \min_{s_j \in \text{NAS}} \left\{\text{LBSU}_j\right\}, \end{aligned} \tag{22}$$

and |NAS| is the number of activated mobile edge servers.

(iv) Load balancing for server-data (LBSD) is the average load balance among the activated MESs based on the number of received data items from mobile users by each activated MES. Assume that the load balance for each activated MES s_j in terms of received data items is denoted as LBSD_j. The average load balance for server-data, avg LBSD, is defined as follows:

$$\text{avg LBSD} = \frac{\text{Max LBSD} - \text{Min LBSD}}{|\text{NAS}|}, \tag{23}$$

where

$$\begin{aligned} \text{Max LBSD} &= \max_{s_j \in \text{NAS}} \left\{\text{LBSD}_j\right\}, \\ \text{Min LBSD} &= \min_{s_j \in \text{NAS}} \left\{\text{LBSD}_j\right\}. \end{aligned} \tag{24}$$

Note that lower values of avgLBSU and avgLBSD are better for satisfying the load balance on the activated servers.

3.3. Effect of the Number of Data Types. Here, the effects of different number of data types are studied and discussed for all strategies when the number of mobile edge servers is 20, and the number of mobile users is 270 in case of epfl/mobility set, and 130 in case of geolife trajectory and roma/taxi sets. Figures 5(a)–5(c) show the number of activated mobile edge servers for epfl/mobility, geolife trajectory, and roma/taxi, respectively. As illustrated in Figures 5(a)–5(c), the number of activated servers grows as the number of data types grows for most strategies. Additionally, the proposed strategies, OTACS and TTACS, achieved reasonable values which are not very low or very high. This is because OTACS and TTACS use the k-means clustering approach to reduce the number of candidate servers, and it can select the appropriate number of activated servers based on their fitness function. Additionally, the number of activated servers of TTACS is greater than OTACS for most values of the number of data types. This is because TTACS uses the ant colony approach in its two tiers for improving the load balancing of the activated servers, while OTACS uses ant colony in one tier only.

Figures 6(a)–6(c) show the achieved total cost for epfl/mobility, geolife trajectory, and roma/taxi, respectively. As illustrated in Figures 6(a)–6(c), the achieved total cost increases as the number of data types increases for most strategies. Additionally, the proposed strategies OTACS and TTACS achieved the lowest total cost among all strategies. This is because OTACS and TTACS use the *k*-means clustering approach and ant colony approach with a fitness function that depends on minimizing the facility and service cost together.

Figures 7(a)–7(c) show the average load balancing for sever-user (avgLBSU) for epfl/mobility, geolife trajectory, and roma/taxi, respectively. As illustrated in Figures 7(a)–7(c), avgLBSU decreases as the number of data types increases for most strategies except the DIS strategy. This is because when the number of data items increases, the number of activated servers increases, and the load balancing gets distributed among them. Additionally, LBSU values of OTACS and TTACS are lower than LBSU values of other strategies. This is beacuse OTACS and TTACS can distribute the server load in terms of users by using the *k*-

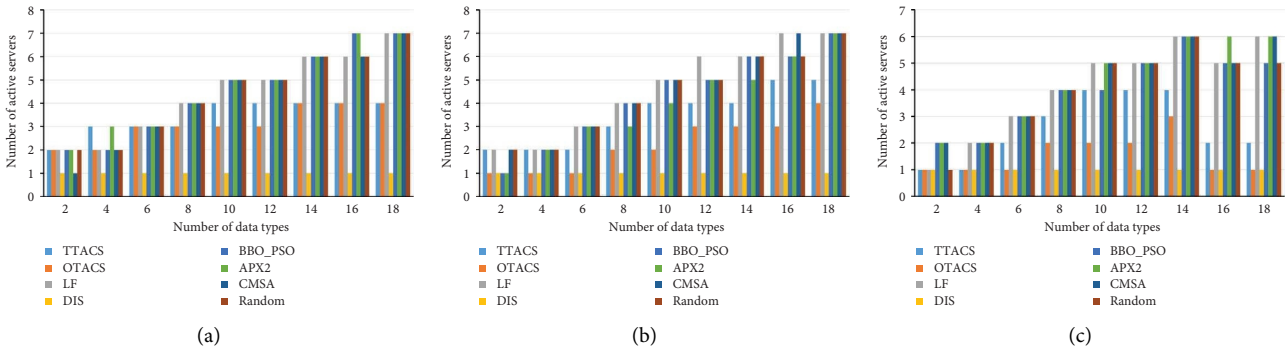

Figure 5: Number of activated servers in terms of data types: (a) epfl/mobility set, (b) geolife trajectory set, and (c) roma/taxi set.

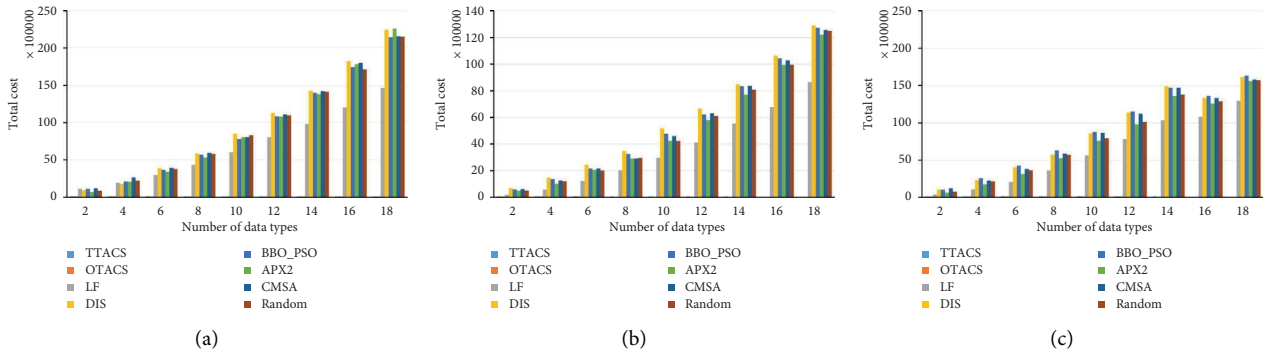

Figure 6: Total cost in terms of data types: (a) epfl/mobility set, (b) geolife trajectory set, and (c) roma/taxi set.

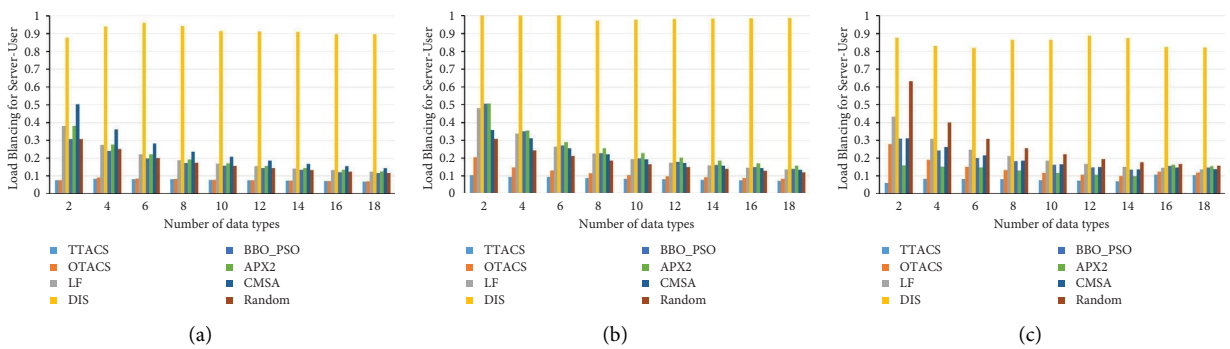

Figure 7: Load balancing for server-user (LBSU) in terms of data types: (a) epfl/mobility set, (b) geolife trajectory set, and (c) roma/taxi set.

mean clustering and ant colony approach. In the case of DIS strategy, the avgLBSU values are less affected by changing the number of data types. This because DIS uses only the travelling distance to select the servers to be activated.

Figures 8(a)–8(c) show the average load balancing for sever-data (avgLBSD) for epfl/mobility, geolife trajectory, and roma/taxi, respectively. As illustrated in Figures 8(a)–8(c), avgLBSD decreases as the number of data types grows for most strategies except the DIS strategy. This is because when the number of data items increases, the number of activated servers increases, and the load balancing gets distributed among them. Additionally, LBSU values of OTACS and TTACS are lower than LBSU values of other strategies. This is because OTACS and TTACS can distribute the server load in terms of users by using the k-mean clustering and ant colony approach. In the case of DIS strategy, the avgLBSD values are less affected by changing the number of data types. This is because DIS uses only the travelling distance to select the servers to be activated.

3.4. Effect of the Number of Servers. Here, the impact of different number of servers are studied and discussed for all strategies when the number of data types is 6 and the mobile users number is 270 in the case of epfl/mobility set and 130 in the case of geolife trajectory and roma/taxi sets. Figures 9(a)–9(c) illustrate the number of activated mobile edge servers for epfl/mobility, geolife trajectory, and roma/taxi, respectively. As illustrated in Figures 9(a)–9(d), the

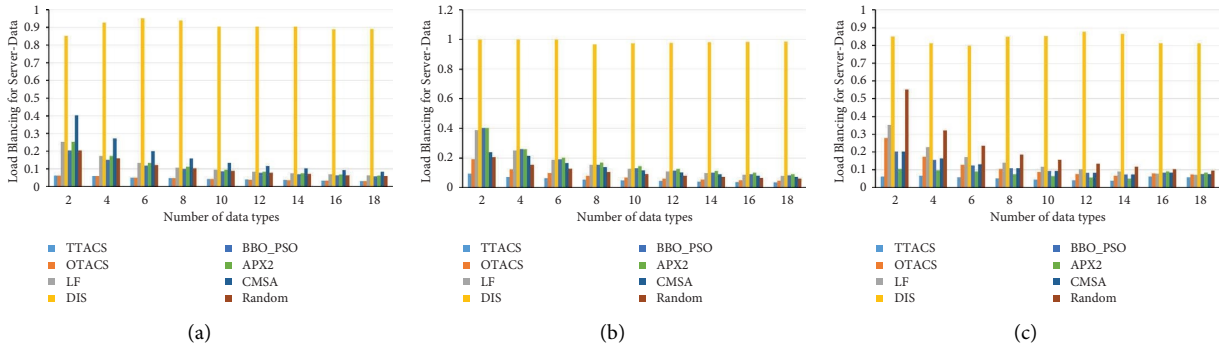

Figure 8: Load balancing for server-data (LBSD) in terms of data types: (a) epfl/mobility set, (b) geolife trajectory set, and (c) roma/taxi set.

number of activated servers is less affected as the number of candidate servers grows for most strategies. Additionally, the proposed strategies, OTACS and TTACS, achieved higher values than other strategies, while in the case of roma/taxi, OTACS and TTACS achieved reasonable values which are not very low or very high. This is because OTACS and TTACS use the k-means clustering approach to reduce the number of candidate servers, and it can select the appropriate number of activated servers based on their fitness function.

Figures 10(a)–10(c) show the achieved total cost for epfl/mobility, geolife trajectory, and roma/taxi, respectively. As illustrated in Figures 10(a)–10(c), the achieved total cost is less affected as the number of candidate servers grows for most strategies. In addition, the proposed strategies, OTACS, TTACS, and APX2, achieved the lowest total cost among all strategies. This is because OTACS, TTACS, and APX2 take into account facility and service costs. Additionally, OTACS and TTACS use the k-means clustering approach and ant colony approach with a fitness function that depends on minimizing the facility and service cost together, and APX2 uses a relaxation-based approximation algorithm.

Figures 11(a)–11(c) show the average load balancing for sever-user (avgLBSU) for epfl/mobility, geolife trajectory, and roma/taxi, respectively. As illustrated in Figures 11(a)–11(c), avgLBSU is less affected as the number of candidate servers grows for most strategies. This is beacuse when the number of candidate servers grows, the number of activated servers is less affected and the load balancing will be distributed among them. Additionally, the LBSU values of OTACS and TTACS are lower than LBSU values of other strategies. This is because OTACS and TTACS can distribute the server load in terms of users by using the k-mean clustering and ant colony approach.

Figures 12(a)–12(c) show the average load balancing for sever-data (avgLBSD) for epfl/mobility, geolife trajectory, and roma/taxi, respectively. As illustrated in Figures 12(a)–12(c), avgLBSD is less affected as the number of candidate servers grows for most strategies. This is because when the number of candidate servers increases, the number of activated servers is less affected and the load balancing will be distributed among them. Additionally, LBSU values of OTACS and TTACS are lower than LBSU values of other strategies. This is because OTACS and TTACS can distribute the server load in terms of users by using the k-mean clustering and ant colony approach.

3.5. Effect of the Number of Users. Here, the impact of different number of users are studied and discussed for all strategies when the number of data types is 6 and the number of mobile edge servers is 20 for epfl/mobility, geolife trajectory, and roma/taxi sets. Figures 13(a)–13(c) show the number of activated mobile edge servers for epfl/mobility, geolife trajectory, and roma/taxi, respectively. As illustrated in Figures 13(a)–13(c), the number of activated servers is less affected as the number of candidate servers grows for most strategies. Additionally, the proposed strategies, OTACS and TTACS, achieved higher values than other strategies. While in the case of roma/taxi, OTACS and TTACS achieved reasonable values which are not very low or very high. This is because OTACS and TTACS use the k-means clustering approach to reduce the number of candidate servers and it can select the appropriate number of activated servers based on their fitness function.

Figures 14(a)–14(c) show the achieved total cost for epfl/mobility, geolife trajectory, and roma/taxi, respectively. As illustrated in Figures 14(a)–14(c), the achieved total cost grows as the number of users grows for most strategies. Additionally, the proposed strategies OTACS, TTACS, and APX2 achieved the lowest total cost among all strategies. This is because OTACS, TTACS, and APX2 take into account facility and service costs. Additionally, OTACS and TTACS use the k-means clustering approach and ant colony approach with a fitness function that depends on minimizing the facility and service cost together and APX2 uses a relaxation-based approximation algorithm.

Figures 15(a)–15(c) show the average load balancing for sever-user (avgLBSU) for epfl/mobility, geolife trajectory, and roma/taxi, respectively. As illustrated in Figures 15(a)–15(c), avgLBSU is less affected as the number of users increases for most strategies. This is because when the number of users increases, the number of the activated servers are less affected and the load balancing will be distributed among them. Additionally, LBSU values of OTACS and TTACS are lower than LBSU values of other strategies. This is because OTACS and TTACS can distribute the server load

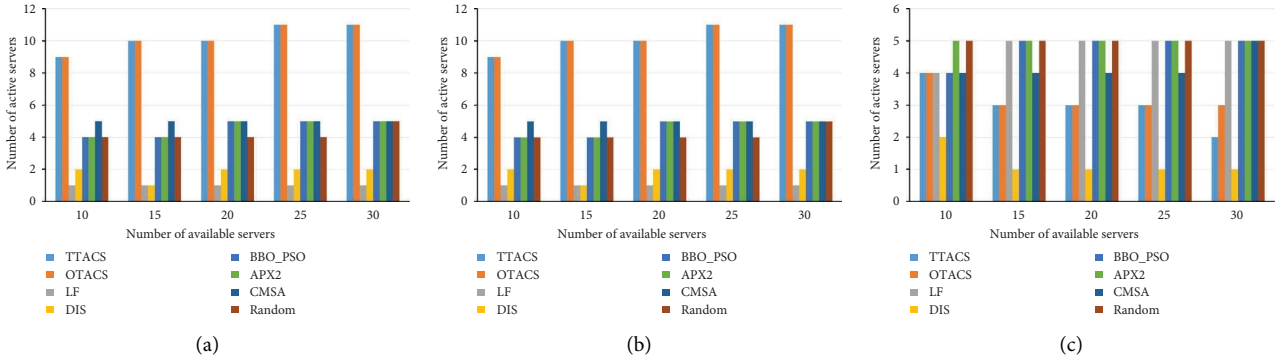

Figure 9: Number of activated servers in terms of candidate servers: (a) epfl/mobility set, (b) geolife trajectory set, and (c) roma/taxi set.

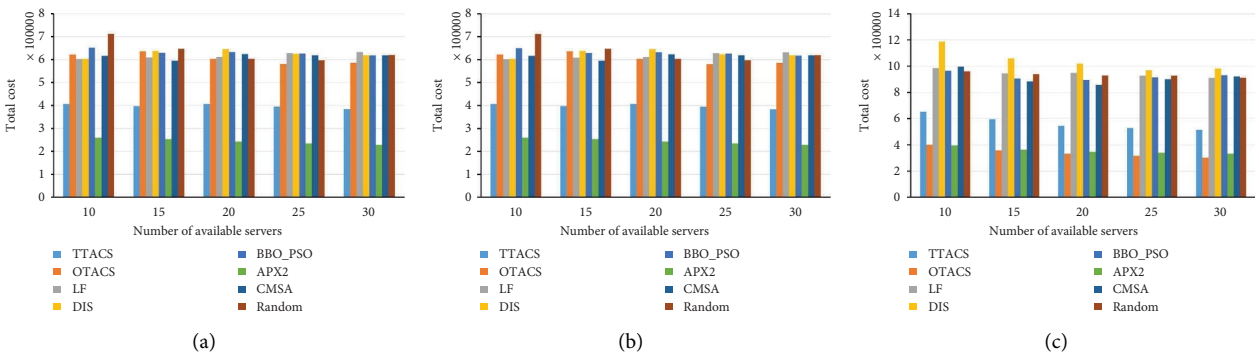

Figure 10: Total cost in terms of candidate servers: (a) epfl/mobility set, (b) geolife trajectory set, and (c) roma/taxi set.

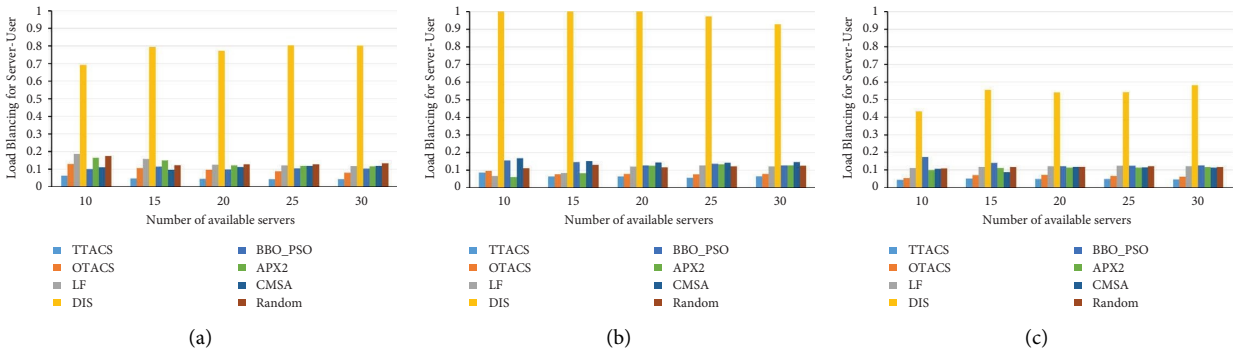

Figure 11: Load balancing for server-user (LBSU) in terms of candidate servers: (a) epfl/mobility set, (b) geolife trajectory set, and (c) roma/taxi set.

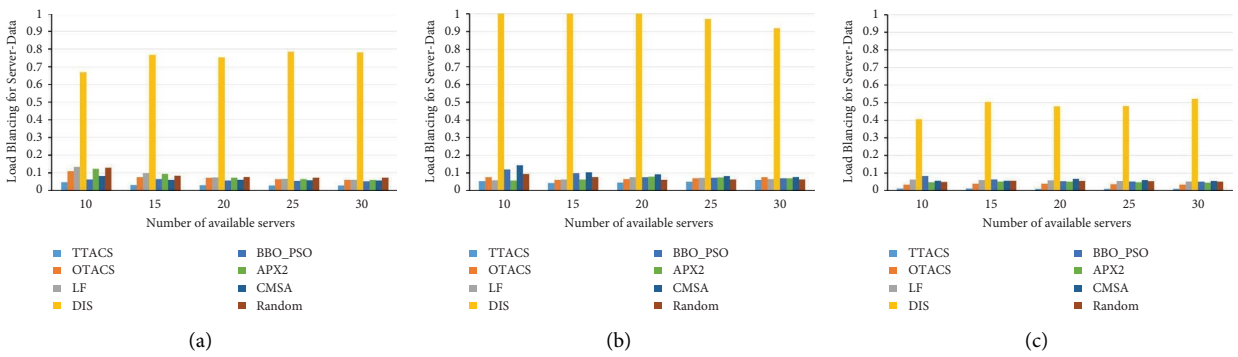

Figure 12: Load balancing for server-data (LBSD) in terms of candidate servers: (a) epfl/mobility set, (b) geolife trajectory set, and (c) roma/taxi set.

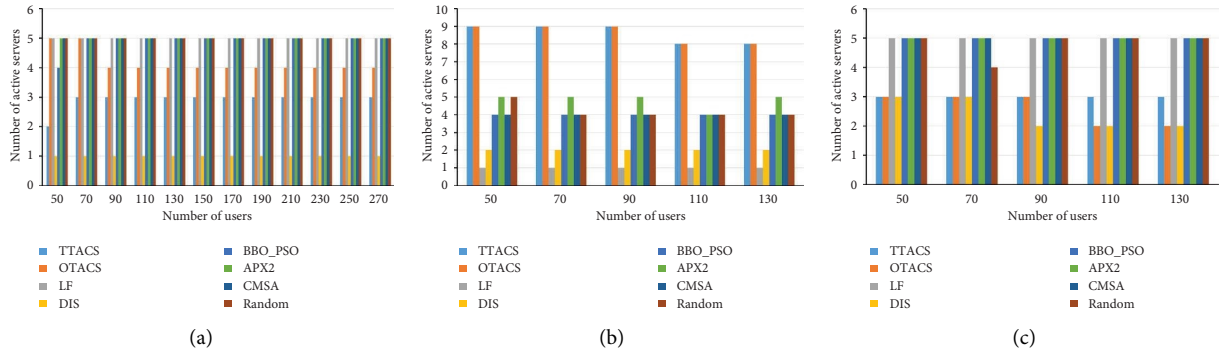

Figure 13: Number of activated servers in terms of users: (a) epfl/mobility set, (b) geolife trajectory set, and (c) roma/taxi set.

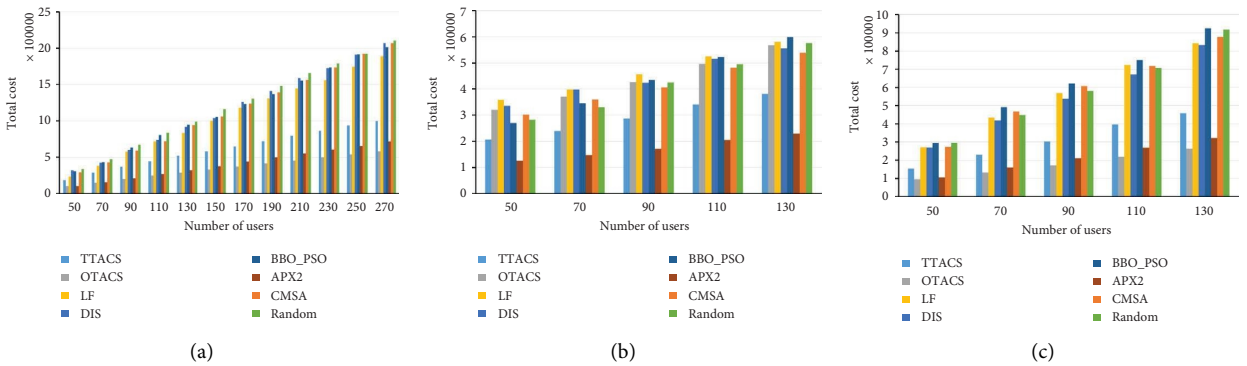

Figure 14: Total cost in terms of candidate servers: (a) epfl/mobility set, (b) geolife trajectory set, and (c) roma/taxi set.

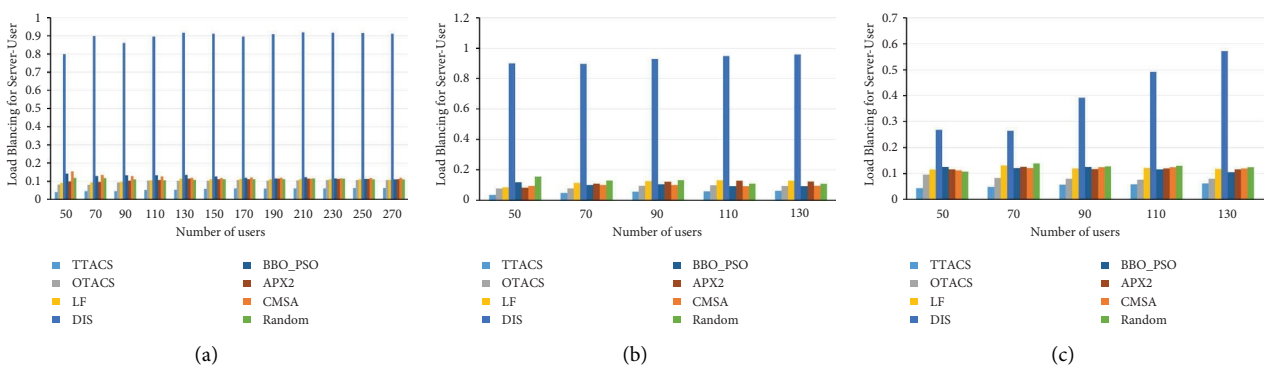

Figure 15: Load balancing for server-user (LBSU) in terms of users: (a) epfl/mobility set, (b) geolife trajectory set, and (c) roma/taxi set.

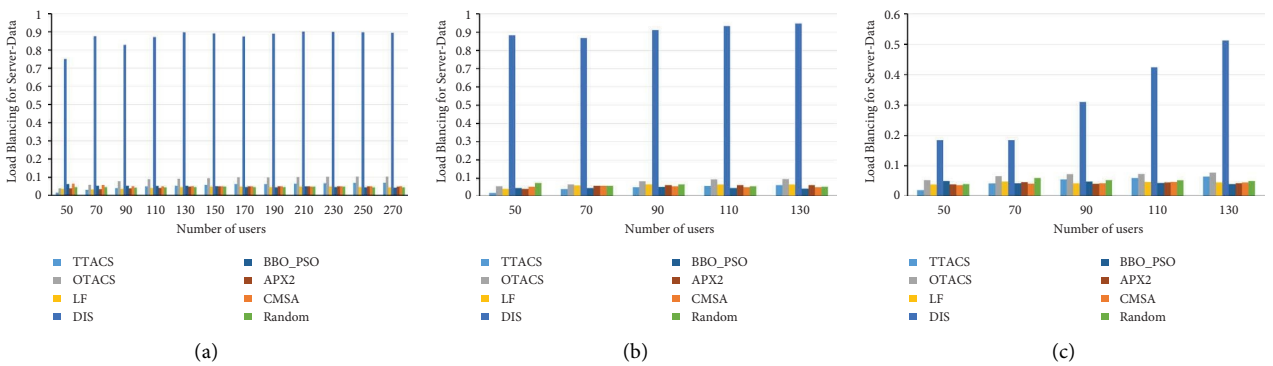

Figure 16: Load balancing for server-data (LBSD) in terms of users: (a) epfl/mobility set, (b) geolife trajectory set, and (c) roma/taxi set.

in terms of users by using the k-mean clustering and ant colony approach.

Figures 16(a)–16(c) show the average load balancing for sever-data (avgLBSD) for epfl/mobility, geolife trajectory, and roma/taxi, respectively. As illustrated in Figures 16(a)–16(c), avgLBSD is less affected as the number of users grows for most strategies. This is because when the number of users increases, the number of activated servers is less affected and the load balancing will be distributed among them. Additionally, LBSU values of OTACS and TTACS are lower than LBSU values of other strategies. This is because OTACS and TTACS can distribute the server load in terms of users by using the k-mean clustering and ant colony approach.

In summary, the results of these conducted simulations show that in the first case "changing number of data types," the proposed methods OTACS and/or TTACS outperform all existing methods in terms of total cost and load balancing for server-user and server-data by which they achieved lower values than existing methods. In the second case "changing number of servers" and the third case "changing number of users," the proposed methods OTACS and/or TTACS outperform all existing methods in terms of total cost except APX2. However, they outperform all existing methods in terms of load balancing for server-user and server-data by which they achieved lower values than existing methods. So, the proposed methods are better than APX2 because they can guarantee the load balance among candidate servers but APX2 cannot.

4. Conclusion

In this paper, two edge-server location strategies are proposed for minimizing the facility and service cost in mobile crowdsensing. These two strategies are called one-tier ant colony clustering-based strategy (OTACS) and two-tier ant colony clustering-based strategy (TTACS). Each proposed strategy uses a clustering method for dividing the set of mobile users with data items into clusters. OTACS uses the ant colony approach in the first tier to select a mobile edge server for each data type in each cluster. Then, it merges all the selected sets of mobile edge servers and uses a simple heuristic method in the second tier to reallocate each data type to its appropriate mobile edge server, while TTACS uses an ant colony approach in the two tier to do the selection and reallocating processes. The proposed strategies were compared with six of the existing strategies. The conducted simulations were based on widely used data sets in the real world: ep/mobility, roma/taxi, and geolife trajectory. The simulation results show that the proposed strategies achieve better performance than the existing methods in terms of service cost, facility cost, and load balancing for server-user server-data distribution. In the future work, new issues will be considered, such as the energy consumption by mobile users, using different clustering algorithms and applying different MCS scenarios.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This project was funded by the Deanship of Scientific Research (DSR), King Abdul-Aziz University, Jeddah, under grant no. DF-868-130-1441. The authors, therefore, acknowledge DSR technical and financial support.

References

[1] R. K. Ganti, F. Ye, and H. Lei, "Mobile crowd sensing: current state and future challenges," *IEEE Communications Magazine*, vol. 49, no. 11, pp. 32–39, 2011.

[2] E. Wang, Y. Yang, J. Wu, W. Liu, and X. Wang, "An efficient prediction based user recruitment for mobile crowdsensing," *IEEE Transactions on Mobile Computing*, vol. 17, no. 1, pp. 16–28, 2018.

[3] W. Liu, Y. Yang, E. Wang, and J. Wu, "Dynamic user recruitment with truthful pricing for mobile crowdsensing," in *Proceedings of the IEEE INFOCOM 2020-IEEE Conference on Computer Communications*, pp. 1113–1122, Toronto, Canada, July 2020.

[4] E. Wang, M. Zhang, X. Cheng et al., "Deep learning-enabled sparse industrial crowdsensing and prediction," *IEEE Transactions on Industrial Informatics*, vol. 17, no. 9, pp. 6170–6181, 2021.

[5] J. Wang, L. Wang, Y. Wang, D. Zhang, and L. Kong, "Task allocation in mobile crowd sensing: state-of-the-art and future opportunities," *IEEE Internet of Things Journal*, vol. 5, no. 5, pp. 3747–3757, 2018.

[6] Y. Yang, W. Liu, E. Wang, and J. Wu, "A prediction-based user selection framework for heterogeneous mobile crowdsensing," *IEEE Transactions on Mobile Computing*, vol. 18, no. 11, pp. 2460–2473, 2019.

[7] Z. Han, J. Liao, Q. Qi, H. Sun, and J. Wang, "Radio environment map construction by kriging algorithm based on mobile crowd sensing," *Wireless Communications and Mobile Computing*, vol. 2019, Article ID 4064201, 12 pages, 2019.

[8] K. Banti, M. Louta, and K. George, "Parkcar: a smart roadside parking application exploiting the mobile crowdsensing paradigm," in *Proceedings of the 2017 8th International Conference on Information, Intelligence, Systems and Applications (IISA)*, pp. 1–6, Larnaca, Cypru, August 2017.

[9] X. Li and D. W. Goldberg, "Toward a mobile crowdsensing system for road surface assessment," *Computers, Environment and Urban Systems*, vol. 69, pp. 51–62, 2018.

[10] D. Sabella, A. Vaillant, P. Kuure, U. Rauschenbach, and F. Giust, "Mobile-edge computing architecture: the role of mec in the internet of things," *IEEE Consumer Electronics Magazine*, vol. 5, no. 4, pp. 84–91, 2016.

[11] K. T. Kim, C. Joe-Wong, and M. Chiang, "Coded edge computing," in *Proceedings of the IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*, pp. 237–246, Toronto, Canada, July 2020.

[12] J. Zhang, L. Zhou, F. Zhou et al., "Computation-efficient offloading and trajectory scheduling for multi-uav assisted mobile edge computing," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 2, pp. 2114–2125, 2020.

[13] X. Li, W. Li, Q. Yang, W. Yan, and A. Y. Zomaya, "Edge-computing-enabled unmanned module defect detection and diagnosis system for large-scale photovoltaic plants," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 9651–9663, 2020.

[14] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile edge computing: a survey," *IEEE Internet of Things Journal*, vol. 5, no. 1, pp. 450–465, 2018.

[15] D. Luan, E. Wang, W. Liu, Y. Yang, and J. Wu, "Minimum-cost edge-server location strategy in mobile crowdsensing," *IEEE Transactions on Network and Service Management*, vol. 18, no. 3, pp. 3375–3388, 2021.

[16] L. Pu, X. Chen, G. Mao, Q. Xie, and J. Xu, "Chimera: an energy-efficient and deadline-aware hybrid edge computing framework for vehicular crowdsensing applications," *IEEE Internet of Things Journal*, vol. 6, no. 1, pp. 84–99, 2019.

[17] J. Ni, K. Zhang, Y. Yu, X. Lin, and X. S. Shen, "Providing task allocation and secure deduplication for mobile crowdsensing via fog computing," *IEEE Transactions on Dependable and Secure Computing*, vol. 17, no. 3, pp. 581–594, 2020.

[18] X. Xia, Y. Zhou, J. Li, and R. Yu, "Quality-aware sparse data collection in mec-enhanced mobile crowdsensing systems," *IEEE Transactions on Computational Social Systems*, vol. 6, no. 5, pp. 1051–1062, 2019.

[19] J. Xiong, X. Chen, Q. Yang, L. Chen, and Z. Yao, "A task-oriented user selection incentive mechanism in edge-aided mobile crowdsensing," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 4, pp. 2347–2360, 2020.

[20] G. Zhao, H. Xu, Y. Zhao, C. Qiao, and L. Huang, "Offloading dependent tasks in mobile edge computing with service caching," in *Proceedings of the IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*, pp. 1997–2006, Toronto, Canada, July 2020.

[21] L. Tong, Y. Li, and W. Gao, "A hierarchical edge cloud architecture for mobile computing," in *Proceedings of the IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*, pp. 1–9, San Francisco, CA, USA, April 2016.

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

[23] T. X. Tran and D. Pompili, "Joint task offloading and resource allocation for multi-server mobile-edge computing networks," *IEEE Transactions on Vehicular Technology*, vol. 68, no. 1, pp. 856–868, 2019.

[24] W. Gong, X. Huang, G. Huang, B. Zhang, and L. Cheng, "Data offloading for mobile crowdsensing in opportunistic social networks," in *Proceedings of the 2019 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6, Waikoloa, HI, USA, December 2019.

[25] L. Wang, D. Zhang, Z. Yan, H. Xiong, and B. Xie, "effSense: a novel mobile crowd-sensing framework for energy-efficient and cost-effective data uploading," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 45, no. 12, pp. 1549–1563, 2015.

[26] Y. Sei and A. Ohsuga, "Differential private data collection and analysis based on randomized multiple dummies for untrusted mobile crowdsensing," *IEEE Transactions on Information Forensics and Security*, vol. 12, no. 4, pp. 926–939, 2017.

[27] J. Chen, H. Ma, Z. Dong, and L. Liu, "Correlated differential privacy protection for mobile crowdsensing," *IEEE Transactions on Big Data*, vol. 7, no. 4, pp. 784–795, 2021.

[28] L. Ma, X. Liu, Q. Pei, and Y. Xiang, "Privacy-preserving reputation management for edge computing enhanced mobile crowdsensing," *IEEE Transactions on Services Computing*, vol. 12, no. 5, pp. 786–799, 2019.

[29] M. Marjanović, A. Antonić, and I. P. Žarko, "Edge computing architecture for mobile crowdsensing," *IEEE Access*, vol. 6, pp. 10662–10674, 2018.

[30] L. Liu, L. Wang, and X. Wen, "Joint network selection and traffic allocation in multi-access edge computing-based vehicular crowdsensing," in *Proceedings of the IEEE INFOCOM 2020 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS)*, pp. 1184–1189, Toronto, Canada, July 2020.

[31] P. Bellavista, M. Cilloni, G. D. Modica, P. C. M. Picone, and M. Solimando, "An edge-based distributed ledger architecture for supporting decentralized incentives in mobile crowdsensing," in *Proceedings of the 2020 20th IEEE/ACM International Symposium on Cluster, Cloud and Internet Computing (CCGRID)*, pp. 781–787, Melbourne, Australia, May 2020.

[32] C. Ying, H. Jin, X. Wang, and Y. Luo, "Chaste: incentive mechanism in edge-assisted mobile crowdsensing," in *Proceedings of the 2020 17th Annual IEEE International Conference on Sensing, Communication, and Networking (SECON)*, pp. 1–9, Como, Italy, June 2020.

[33] L. Liu, X. Wen, L. Wang, Z. Lu, W. Jing, and Y. Chen, "Incentive-aware recruitment of intelligent vehicles for edge-assisted mobile crowdsensing," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 10, pp. 12085–12097, 2020.

[34] F. A. Chudak and D. P. Williamson, "Improved approximation algorithms for capacitated facility location problems," in *Integer Programming and Combinatorial Optimization*, G. . Cornuéjols, R. E. Burkard, and G. J. Woeginger, Eds., pp. 99–113, Springer Berlin Heidelberg, Berlin, Heidelberg, 1999.

[35] M. T. Melo, S. Nickel, and F. Saldanha da Gama, "Largescale models for dynamic multicommodity capacitated facility location," Technical Report 58, Fraunhofer Institute for Industrial Mathematics, Kaiserslautern, Germany, 2003.

[36] S. R. Shariff, N. H. Moin, and M. Omar, "Location allocation modeling for healthcare facility planning in Malaysia," *Computers & Industrial Engineering*, vol. 62, no. 4, pp. 1000–1010, 2012.

[37] J. Zhang, "Approximating the two-level facility location problem via a quasi-greedy approach," *Mathematical Programming*, vol. 108, no. 1, pp. 159–176, 2006.

[38] D. B. Shmoys, V. Tardos, and K. Aardal, "Approximation algorithms for facility location problems," in *Proceedings of the 29th Acm Symposium on Theory of Computing*, El Paso, TX, USA, May 1997.

[39] A. Mohammad Nezhad, H. Manzour, and S. Salhi, "Lagrangian relaxation heuristics for the uncapacitated single-source multi-product facility location problem," *International Journal of Production Economics*, vol. 145, no. 2, pp. 713–723, 2013.

[40] Z. Svitkina, "Lower-bounded facility location," *ACM Transactions on Algorithms*, vol. 6, no. 4, pp. 1–16, 2010.

[41] Z. Friggstad and M. R. Salavatipour, "Minimizing movement in mobile facility location problems," in *Proceedings of the*

2008 49th Annual IEEE Symposium on Foundations of Computer Science, pp. 357–366, Philadelphia, PA, USA, October 2008.

[42] M. Charikar and S. Guha, "Improved combinatorial algorithms for the facility location and k-median problems," in *Proceedings of the 40th Annual Symposium on Foundations of Computer Science (Cat. No.99CB37039)*, pp. 378–388, New York City, NY, USA, October 1999.

[43] R. Fleischer, J. Li, S. Tian, and H. Zhu, "Non-metric multicommodity and multilevel facility location," in *Algorithmic Aspects in Information and Management*, S. W. Cheng and K. P. Chung, Eds., pp. 138–148, Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[44] R. Ravi and A. Sinha, "Multicommodity facility location," in *Proceedings of the 15th Acm-Siam Symposium on Discrete Algorithms*, pp. 342–349, New Orleans, LA, USA, January 2004.

[45] E. L. Mooney, Y. Almoghathawi, and K. Barker, "Facility location for recovering systems of interdependent networks," *IEEE Systems Journal*, vol. 13, no. 1, pp. 489–499, 2019.

[46] A. A. Abin, "Querying beneficial constraints before clustering using facility location analysis," *IEEE Transactions on Cybernetics*, vol. 48, no. 1, pp. 312–323, 2018.

[47] L. P. Meng, Q. Kang, C. F. Han, and M. C. Zhou, "Determining the optimal location of terror response facilities under the risk of disruption," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 2, pp. 476–486, 2018.

[48] T. Niknam, "A new approach based on ant colony optimization for daily volt/var control in distribution networks considering distributed generators," *Energy Conversion and Management*, vol. 49, no. 12, pp. 3417–3424, 2008.

[49] M. Dorigo and C. Blum, "Ant colony optimization theory: a survey," *Theoretical Computer Science*, vol. 344, no. 2-3, pp. 243–278, 2005.

[50] P. Berkhin, *A Survey of Clustering Data Mining Techniques*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2006.

[51] L. Bracciale, M. Bonola, P. Loreti, G. Bianchi, and R. Amici, "Crawdad dataset roma/taxi," 2014, https://crawdad.org/roma/taxi/20140717/.

[52] M. Piórkowski, S.-D. Natasa, and G. Matthias, "CRAWDAD wireless network data archive," 2009, https://crawdad.org/epfl/mobility/20090224.

[53] Y. Zheng, L. Zhang, X. Xie, and W.-Y. Ma, "Mining interesting locations and travel sequences from gps trajectories," in *Proceedings of the 18th International Conference on World Wide Web*, pp. 791–800, ACM, Madrid Spain, April 2009.