Research Article

A Mobility-Aware and Sociality-Associate Computation Offloading Strategy for IoT

Yanfei Lu,¹ Zengzi Chen ,¹ Qinghe Gao ,¹ Tao Jing ,¹ and Jin Qian ²

¹School of Electronic and Information Engineering, Beijing Jiaotong University, Beijing, China

²College of Computer Science & Technology, Taizhou University, Jiangsu, China

Correspondence should be addressed to Qinghe Gao; qhgao@bjtu.edu.cn

Received 13 March 2021; Revised 31 May 2021; Accepted 9 June 2021; Published 28 June 2021

Academic Editor: Yan Huang

Copyright © 2021 Yanfei Lu et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Mobile edge computing, a promising paradigm, brings services closer to a user by leveraging the available resources in an edge network. The crux of MEC is to reasonably allocate resources to satisfy the computing requirements of each node in the network. In this paper, we investigate the service migration problem of the offloading scheme in a power-constrained network consisting of multiple mobile users and fixed edge servers. We propose an affinity propagation-based clustering-assisted offloading scheme by taking into account the users' mobility prediction and sociality association between mobile users and edge servers. The clustering results provide the candidate edge servers, which greatly reduces the complexity of observing all edge servers and decreases the rate of service migration. Besides, the available resource of candidate edge servers and the channel conditions are considered to optimize the offloading scheme to guarantee the quality of service. Numerical simulation results demonstrate that our offloading strategy can enhance the data processing capability of power-constrained networks and reach computing load balance.

1. Introduction

The arrival and evolution of the 5G delivers a transformative solution to an ultimate high-quality end-user experience. Such a high-speed, high-capacity, low-latency 5G network can well meet the demands of increasing data-intensive applications in the current Internet of Things (IoT) and Artificial Intelligence (AI) era. The numerous applications require high-speed internet connectivity and high computation power, which is not possible in a mobile device with limited memory and storage capacity [1–3]. In such situations, it is feasible to transfer resource-intensive tasks to external platforms like cloud, grid, and edge servers. This process is known as task offloading, which decides when and how a task should be offloaded to external platforms to execute.

During the development of 5G, the mobile edge computing (MEC) technique has been playing an important role [4, 5]. Massive smart edge devices in various IoT systems hide huge distributed computing capabilities [6, 7]. The reasonable utilization of distributed capability can reduce the amount and frequency of information transferring to a dis-

tant centralized cloud server, which can greatly decrease transfer latency and omit the cloud server processing latency. Although existing research can reduce the delay in some scenarios, many severe problems remain to be solved. Especially, how to migrate services as mobile users move is a critical problem. If users move far away from the MEC server that is responding to their request, this could result in significant quality of service (QoS) and quality of experience (QoE) degradation and service interruption due to long transmission latency of the offloaded task [8]. To ensure the continuity of service, services need to be migrated to nearby MEC servers that can cover the user's current location when it moved out of previous service coverage.

The migration of user-generated data in edge networks is involved in transmission costs, the mobility of users, transmission resources, etc. Thus, an efficient edge server selection algorithm is needed to select the optimal target edge server. In general, two factors should be taken into account: users' trajectory and QoS utility. On the one hand, existing research works rarely explore users' trajectory data and the prediction of their movement and adopt a random mobility model

instead [9]. However, users' mobility pattern (e.g. direction and velocity) has a significant influence on the construction of the candidate edge server set and the users' trajectory data can be used to predict users' movement. On the other hand, existing literatures pay less attention on the effect of QoS utility (network latency, energy consumption, and cost) on the selection of edge servers in service migration and, therefore, hardly select the edge server with the highest QoS utility [10–13]. Without considering users' trajectory data and QoS utility, the accuracy of edge server selection and the efficiency of service migration decrease. To develop a QoS-aware algorithm to improve edge server selection, we should overcome the problems such as how to integrate user's trajectory data and QoS utility into the server selection algorithm.

In this paper, we investigate an offloading scheme by considering users' trajectory data, sociality associations, and QoS parameters. First, we employ a Kalman filtering algorithm to do mobility prediction of users' trajectory; then, we compute the sociality association between mobile users and edge servers by using the history connection relationship. By using the mobility prediction and the social association results as input parameters of an affinity propagation algorithm, we propose a mobility-aware and sociality-associate clustering algorithm (MASACA), where MEC servers are divided into candidate sets associated with different users. Moreover, we devise a QoS utility function for each MEC server based on their available computation resource and the channel links' quality, which is used to determine the appropriate MEC server for a user to offload task. The main contributions of this paper are summarized as follows:

- (1) A mobility model of mobile users is considered, which is combined with a Kalman filter to make prediction about the location of users
- (2) The sociality association is taken into account, which guarantees the continuity of service
- (3) Clustering has been shown to achieve efficient and reliable management and reduce data congestion. We construct an AP-based algorithm to obtain stable clusters considering the trajectory data and the sociality association of mobile users

The paper is organized as follows: Section 2 gives a review of related works in the literature. Section 3 presents the system model and problem formulation. Section 4 gives the details of AP-MASACA. The QoS utility function encompassing outage probability and average coverage is derived in Section 5. Section 6 presents the performance of the proposed scheme by comparing with existing offloading schemes. And a summary is provided in Section 7.

2. Related Works

In order to optimize the delay and energy consumption of mobile devices, a computational offload strategy is adopted in mobile edge computing [14]. Mobility is one key challenging topic in MEC systems, which has effects on decisions in several domains such as caching, connected vehicles, and

especially computation offloading [15]. However, the mobility of mobile users and the limited coverage of edge servers can result in significant network performance degradation, dramatic drop in QoS, and even interruption of ongoing edge services [16]. Usually, the types of UE mobility can be categorized as random mobility, short-term predictable mobility, and fully known mobility, depending on whether the future location of the UE is known. Because of the limited resources of servers, the mobility of users, and the low latency requirements of service requests, computing offloading and service migrations are expected to occur in MEC systems regularly. The authors in [17] proposed a glimpse mobility prediction model based on the seq2seq model, which provides useful coarse-grain mobility information for Mobility-Aware Deep Reinforcement Learning training. However, the traditional offloading approaches (e.g., auction-based and game-theory approaches) cannot adjust the strategy according to changing environment when dealing with the mobility problem, in order to keep the system performance for a long time, an online user-centric mobility management scheme is proposed in [18] to maximize the edge computation performance while keeping the energy consumption of user's communication low by using Lyapunov optimization and multiarmed bandit theories. These proposed methods overcome the challenges of mobility in MEC, making contribution in reducing service delays or energy consumption of devices. However, by nature, minimizing the smart devices' energy consumption and tasks' execution time does not always coincide and may conflict in IoT.

Beside delay and energy consumption, the MEC system performance can be further improved by utilizing the recent developments of social networks [13, 19] and energy harvesting methods. Social IoT (SIoT) has been proposed as a promising way to create and maintain the collaborative relations among smart SIoT devices. The idea of SIoT is to build social collaborative networking for smart IoTs to achieve locally distributed data processing according to the rules set by owners [20]. The collaboration between SIoT and MEC is crucial to minimize network communications and computation, where the SIoT collaboration can locally process user requests and further reduce the communications and computation in MEC networks. Nevertheless, the load sharing among SIoT devices, MEC, and remote servers brings about new challenges for the communication and computation trade-off, cross-layer design in SIoT, and forwarding and aggregation trade-off [14, 21]. To solve this issue, Wang et al. in [22] formulated a new optimization problem SIoT Collaborative Group and Device Selection Problem (SCGDSP) and propose Optimal Collaborative Group Selection (OCGS) for a fundamental SCGDSP to find the intrinsic properties of collaborative group construction under the deployment of SIoTs in a grid street map to reduce the running time and transmission delay. However, one of the main challenges of social attribute is the selfish nature of the user, and some users only response what they are interested in and do not care about other people's needs. The authors in [23] proposed a physical-social-based cooperative cache framework to maximizing the social group utility and meet the data request of users. By considering the diversity of

TABLE 1: Summary of notations.

Notations	Meanings
$\mathcal{U}$	The set of mobile users
$\mathcal{S}$	The set of edge servers
$\mathcal{A}_i$	The task set of mobile user U_i
$h_{i,j}$	The channel fading coefficient between U_i and S_j
$d_{i,j}$	The distance between U_i and S_j
N_0	The power of additive white Gaussian noise (AWGN)
$R_{i,j}$	The channel rate between U_i and S_j
$A_{in}^{i,k}$	The input size of task a_k for U_i
$A_{out}^{i,k}$	The size of computation results task a_k for U_i
$A_{com}^{i,k}$	The number of CPU clock cycles required by a_k of U_i
f_{loc}^i	The computation capability of U_i
F_{sev}^j	The computation capability of S_j
$f_{sev}^{i,j,k}$	The actual CPU frequency of S_j for the task a_k of U_i
$T_{loc}^{i,k}$	The time delay for U_i locally executing the task a_k
$T_{in}^{i,j,k}$	The time delay for U_i transmitting the task a_k to S_i
$T_{out}^{i,j,k}$	The time delay for S_i sending computation results of the task a_k to U_i
$T_{sec}^{i,j,k}$	The time delay for U_i offloading task a_k to S_j without service migration
$T_{mig}^{i,k}$	The time delay for migrating computing results of the task a_k of U_i between two edge servers
$T_{sev,m}^{i,j,j,k}$	The time delay for U_i offloading task a_k to S_j with service migration
$T_{sum}^{i,k}$	The time delay for completing the task a_k for U_i
T_{sum}	The total time delay for completing all tasks of all users
P_{loc}^i	The average power when the CPU of U_i is working
$P_{i,0}$	The average power U_i is idle
$P_{i,R}$	The average power U_i is receiving computation results
$P_{i,j}$	The transmission power of U_i for transmitting tasks to S_j
$E_{loc}^{i,k}$	The energy consumption at U_i for locally executing a_k
$E_{sev}^{i,j,k}$	The energy consumption at U_i for offloading a_k to S_j without migration
$E_{sev,m}^{i,j,j,k}$	The energy consumption at U_i for offloading a_k to S_j with migration
E_{sum}^i	The energy consumption at U_i for completing its all tasks
$\lambda_{i,j,k}$	1 if U_i transmits a_k to S_j , and otherwise 0
$\mu_{i,j,j'}^{i,j,k}$	1 if S_j transmits computing results for a_k of U_i to S_j' , and otherwise 0
$\bar{sev}$	The average service ratio

users, the authors of [24, 25] combined MEC with the mobile crowdsensing approach and proposed a social-based method to optimize the share of contents among users by exploiting

their mobility and sociality, which improved the performance of a content-sharing scenario.

There is no common definition of social strength in current researches about SIoT. Jung et al. [26] proposed a model named social strength prediction model, which infers social connections among smart objects and predicts the strength of the connections using the cousage data of the objects by two major components: (1) entropy-based and (2) distance-based social strength computations. These two components capture different properties of co-usages of objects, namely, the diversity and spatiotemporal features, which are all essential factors that contribute to the values of social strength. Wang et al. [27] put forward a method to measure the social strength generally by the available storage capacity and betweenness centrality. In this paper, we propose a definition that utilizes the information entropy to calculate social strength.

3. System Model and Problem Formulation

In this section, we give the system model and computation model of the task offloading and then formulate the offloading problem. For the sake of clarity, we summarize the notations in Table 1.

3.1. System Model. We consider a multiuser MEC model as shown in Figure 1 with a set of mobile users (MUs) denoted as $\mathcal{U} = \{U_1, U_2, \dots, U_i, \dots, N\}$ and a set of edge servers as $\mathcal{S} = \{S_1, S_2, \dots, S_j, \dots, S_M\}$. According to physical positions of each user, they are clustered into different service cells surrounding edge servers. Assume that each MU has K computation tasks. Denote $\mathcal{A}_i = \{a_1, a_2, \dots, a_k, \dots, a_K\}$ as the task set of U_i , where each a_k is independent and decomposable. For each task, a mobile user can proceed it locally or offload it to corresponding edge server within its service cell.

However, the service cell for an MU may change due to the MU's mobility. To guarantee seamless services, service migration is necessary. Specifically, as shown in Figure 2, U_2 initially belonged to the service cell surrounding S_1 and now is within the service cell surrounding S_2 . If it offloaded computing tasks to S_1 before moving out the initial cell and the computation results did not return to U_2 before it enters into the current cell, computation results need to be migrated from S_1 to S_2 , then returned to U_2 .

Thus, there are three computation offloading models: local execution, offloading without service migration, and offloading with service migration. In following sections, we will investigate how to select from these three computation offloading models to minimize the time delay for completing all tasks of all mobile users under the guarantee of successful offloading.

3.2. Computation Offloading Model. Define $A_{in}^{i,k}$ as the input size of task a_k for U_i , $A_{out}^{i,k}$ the size of computation results returned to U_i for a_k , and $A_{com}^{i,k}$ the total number of CPU clock cycles required to complete task a_k .

3.2.1. Local Computing Model. Let f_{loc}^i denote the computation capability of U_i , which is dependent on the intrinsic

FIGURE 1: System model: service cells are clustered by mobile edge computing servers and mobile users.

FIGURE 2: Mobility-aware MEC network.

nature of the MU, i.e., CPU cycles per second. The time $T_{\text{loc}}^{i,k}$ for locally processing the request of task a_k at U_i is expressed as

$$T_{\text{loc}}^{i,k} = \frac{A_{\text{com}}^{i,k}}{f_{\text{loc}}^i}. \quad (1)$$

Assume that the energy consumption per second when the CPU is working is p_{loc}^i . Then, the energy consumption $E_{\text{loc}}^{i,k}$ of local executing the task a_k for U_i can be given as

$$E_{\text{loc}}^{i,k} = p_{\text{loc}}^i T_{\text{loc}}^{i,k}. \quad (2)$$

3.2.2. Offloading without Service Migration. Three phases are needed to offload computation task to an edge server

without service migration: (i) an MU U_i transmits task request to an edge server S_j ; (ii) edge server S_j executes computing task; and (iii) edge server S_j sends computing results to U_i .

Assume that the wireless channel bandwidth between U_i and S_j is $W_{i,j}$. Let $p_{i,j}$ represent the transmission power of U_i for transmitting tasks to S_j . The transmission rate $R_{i,j}$ between U_i and S_j can be calculated by

$$R_{i,j} = W_{i,j} \log_2 \left(1 + \frac{p_{i,j} h_{i,j}^2}{d_{i,j}^\theta N_0} \right), \quad (3)$$

where $h_{i,j}$ represents the transmission channel fading coefficient, $d_{i,j}$ is the distance from U_i to S_j , θ is the standard path loss propagation exponent, and N_0 denotes the power of

additive white Gaussian noise (AWGN). Then, the transmission phase $T_{in}^{i,j,k}$ from U_i to S_j for the task a_k can be given as

$$T_{in}^{i,j,k} = \frac{A_{in}^{i,k}}{R_{i,j}}. \quad (4)$$

The computing capability of an edge server is denoted as F_{sev}^j . Let $f_{sev}^{i,j,k}$ denote the actual CPU frequency when edge server S_j executes the task a_k received from U_i , where $0 \leq f_{sev}^{i,j,k} \leq F_{sev}^j$. Then, the time $T_{com}^{i,j,k}$ for edge server S_j processing the request of task a_k from U_i can be calculated as

$$T_{com}^{i,j,k} = \frac{A_{com}^{i,k}}{f_{sev}^{i,j,k}}. \quad (5)$$

After completing the computation task, edge server S_j needs to send back the computation results to U_i . The time delay for this transmission phase is expressed as

$$T_{out}^{i,j,k} = \frac{A_{out}^{i,k}}{R_{i,j}}. \quad (6)$$

Then, the total time delay for offloading task a_k to an edge server S_j without service migration can be computed as

$$T_{sec}^{i,j,k} = T_{in}^{i,j,k} + T_{com}^{i,j,k} + T_{out}^{i,j,k}. \quad (7)$$

Assume the energy consumption per second for U_i during the idle duration and the receiving phase are denoted as $p_{i,0}$ and $p_{i,R}$, respectively. In total, the energy consumption at U_i for executing the task a_k on MEC sever S_j without service migration can be formulated as

$$E_{sev}^{i,j,k} = p_{i,j} T_{in}^{i,j,k} + p_{i,0} T_{com}^{i,j,k} + p_{i,R} T_{out}^{i,j,k}. \quad (8)$$

3.2.3. Offloading with Service Migration. If U_i moves from a cell surrounding S_j to current cell surrounding S_j' before S_j completes the computing task a_k , one more migration phase is needed to transmit computation results from edge server S_j to current edge server S_j' . The time delay of this migration phase is determined by the transmission rate R_s between these two edge servers as follows:

$$T_{mig}^{i,k} = \frac{A_{out}^{i,k}}{R_s}, \quad (9)$$

where R_s is a fixed value because of optical fiber communications between edge servers. Then, S_j' forwards the computation results to U_i . Similar to (6), the transmission time can be given as

$$T_{out}^{i,j,k} = \frac{A_{out}^{i,k}}{R_{i,j}}, \quad (10)$$

where the transmission rate $R_{i,j}'$ can be calculated in a similar way with (3). Thus, the total time delay $T_{sev,m}^{i,j,j',k}$ for offloading with service migration can be calculated as

$$T_{sev,m}^{i,j,j',k} = T_{in}^{i,j,k} + T_{com}^{i,j,k} + T_{mig}^{i,k} + T_{out}^{i,j',k}. \quad (11)$$

Likewise as (8), the energy consumption at U_i for executing the task a_k by offloading with service migration can be calculated as

$$E_{sev,m}^{i,j,j',k} = p_{i,j} T_{in}^{i,j,k} + p_{i,0} (T_{com}^{i,j,k} + T_{mig}^{i,k}) + p_{i,R} T_{out}^{i,j',k}. \quad (12)$$

3.3. Problem Formulation. For convenience, we define a binary variable $\lambda_{i,j,k}$ to equal to 1 if a_k of U_i is offloaded to S_j and otherwise equal to 0. When a_k of U_i is offloaded to S_j , we define $\mu_{i,j,j',k}$ to be 1 if service migration from S_j to S_j' is necessary and otherwise equal to 0. Thus, the time delay for completing the task a_k for U_i can be expressed as

$$T_{sum}^{i,k} = T_{loc}^{i,k} * \left(1 - \sum_j \lambda_{i,j,k} \right) + T_{sev}^{i,j,k} \lambda_{i,j,k} \left(1 - \sum_{j'} \mu_{i,j,j',k} \right) + T_{sev,m}^{i,j,j',k} \lambda_{i,j,k} \sum_{j'} \mu_{i,j,j',k}. \quad (13)$$

The total time delay T_{sum} for completing all tasks of all users can be given by

$$T_{sum} = \sum_{i=1}^N \sum_{k=1}^K T_{sum}^{i,k}. \quad (14)$$

In summary, the problem of minimizing the total time delay by optimizing the transmission power matrix $\mathbf{P} = \{\mathbf{p}_1, \dots, \mathbf{p}_i, \dots, \mathbf{p}_N\} \in \mathbb{R}^{M \times N}$, the offloading matrix $\mathbf{\Lambda} \in \mathbb{R}^{M \times N \times K}$, and the migration matrix $\mathbf{\Gamma} \in \mathbb{R}^{M \times N \times K}$ can be formulated as follows:

$$\begin{aligned} (\mathbf{P1}): \quad & \min_{\mathbf{P}, \mathbf{\Lambda}, \mathbf{\Gamma}} \quad T_{sum} \\ \text{s.t.} \quad & C_1 : 0 \leq f_{sev}^{i,j,k} \leq F_{sev}^j \\ & C_2 : 0 \leq p_{i,j} \leq p_{max}^i \\ & C_3 : \lambda_{i,j,k} \in \{0, 1\}, \forall i, j, k \\ & C_4 : \mu_{i,j,j',k} \in \{0, 1\}, \forall i, j, j', k \\ & C_5 : \sum_j \lambda_{i,j,k} \leq 1, \forall i, k \\ & C_6 : \sum_{j', j' \neq j} \mu_{i,j,j',k} \leq 1, \forall i, k \\ & C_7 : E_{sum}^i \leq E_{max}^i, \quad \forall i, \end{aligned} \quad (15)$$

FIGURE 3: Mobile node wants to get optimized offloading strategies during the movement.

where

$$E_{\text{sum}}^i = \sum_k^K \left[E_{\text{loc}}^{i,k} * \left(1 - \sum_j \lambda_{i,j,k} \right) + E_{\text{sev}}^{i,j,k} \lambda_{i,j,k} \left(1 - \sum_{j'} \mu_{i,j,j'}^{',k} \right) + E_{\text{sev},m}^{i,j,j'} \lambda_{i,j,k} \sum_{j'} \mu_{i,j,j'}^{',k} \right]. \quad (16)$$

Constraint C_1 is the computing capacity limitation of edge server, and constraint C_2 is the power limitation with p_{max}^i being the maximum tolerable transmission power at U_i . Constraints C_3 and C_4 indicate that $\lambda_{i,j,k}$ and $\mu_{i,j,j'}^{',k}$ are binary variables. Constraint C_5 gives the requirement that each task that can be totally offloaded to an edge server or be executed locally as a whole. Constraint C_6 ensures the service migration occurs between one-hop edge servers. The constraint C_7 is the energy constraint at U_i with E_{max}^i being the maximum available energy of U_i . Due to the integer constraints C_3 - C_6 , $\mathbf{P1}$ is a mixed integer programming problem and NP-hard. In order to reduce the complexity of solving this NP-hard problem, in the following sections, we propose a mobility-aware and sociality-associate affinity propagation clustering algorithm, and then design an offloading scheme based on the clustering results.

4. Mobility-Aware and Sociality-Associate Clustering Algorithm (MASACA)

As shown in Figure 3, a MU may meet different edge servers during its movement towards its destination. One of method to decrease the complexity of solving $\mathbf{P1}$ is cutting down the number of candidate edge servers. In this section, we propose an affinity propagation- (AP-) based clustering by utilizing mobility prediction and sociality association results.

The MU can determine to randomly offload its tasks to an adjacent edge server at the moment the tasks arrive. In this

case, service migration may occur due to the movement of MU. When the server completes the computing tasks, the MU may have moved to a remote position and the connection with this server may be lost, leading to the recalculation of tasks and bringing additional overhead. To avoid much service migration and decrease the lost rate of computation results, we use Kalman filtering method to make trajectory prediction. According to the prediction results, the mobile user can select the best timing to offload its tasks to a better edge server.

4.1. Mobility Prediction. The movement of the mobile user is considered as a traditional dynamic system satisfying the following state equation at time $t = aT$:

$$\mathbf{s}_a = \mathbf{A}\mathbf{s}_{a-1} + \mathbf{B}\mathbf{u}_{a-1} + \omega_a, \quad (17)$$

where $\mathbf{s}_a$ is the state vector and equals $\{p_x, p_y, v_x, v_y\}_a$ representing the user's real-time position $\{p_x, p_y\}_a$ and its velocity $\{v_x, v_y\}_a$ in a two-dimensional space; $\mathbf{A}$ is the transition matrix; $\mathbf{u}_{a-1}$ is the input signal from previous time, namely, the moving velocity of the user; $\mathbf{B}$ is the input matrix; ω_a is a deterministic process noise and is usually assumed as a Gaussian noise with $\omega \sim \mathcal{N}(0, \mathbf{Q})$; and $\mathbf{Q}$ is called the process noise covariance matrix. The observation model is denoted as

$$\mathbf{z}_a = \mathbf{H}\mathbf{s}_a + \mathbf{y}_a, \quad (18)$$

where $\mathbf{z}_a$ is the observation vector, $\mathbf{H}$ is the observation matrix, and $\mathbf{y}_a$ is the observation noise and also seen as a Gaussian noise with $\mathbf{y} \sim \mathcal{N}(0, \mathbf{R})$. $\mathbf{R}$ is called the observation noise covariance matrix. Note that ω_a and $\mathbf{y}_a$ describe deviations such as collision, slippage, and friction that caused by objective conditions. Thus, in practice, the process noise covariance $\mathbf{Q}$ and the observation matrix $\mathbf{R}$ might change within each time step and are assumed to be temporally and spatially independent with each other.

According to above analyses, let $\hat{\mathbf{s}}_a$ denote the estimate value of $\mathbf{s}_a$ when the measurement value of $\mathbf{z}_a$ is obtained. The dynamic estimation process includes two steps:

- (i) The time update phase follows the next equations as

$$\begin{cases} \hat{\mathbf{s}}_a^- = \mathbf{A}\hat{\mathbf{s}}_{a-1} + \mathbf{B}\hat{\mathbf{u}}_a, & a \geq 1, \\ \mathbf{G}_a^- = \mathbf{A}\mathbf{G}_{a-1}\mathbf{A}^* + \mathbf{Q}, \end{cases} \quad (19)$$

where $(\cdot)^-$ is the prediction value of a variable and $(\cdot)^*$ and $(\cdot)^{-1}$ represent the transpose matrix and inverse matrix, respectively. $\mathbf{G}$ is the error covariance matrix

- (ii) The measurement update phase is as follows:

$$\begin{cases} \mathbf{K}_a = \mathbf{G}_a^- \mathbf{H}^* (\mathbf{H} \mathbf{G}_a^- \mathbf{H}^* + \mathbf{R})^{-1}, \\ \hat{\mathbf{s}}_a = \hat{\mathbf{s}}_a^- + \mathbf{K}_a (\mathbf{z}_a - \mathbf{H} \hat{\mathbf{s}}_a^-), \\ \mathbf{G}_a = (\mathbf{I} - \mathbf{K}_a \mathbf{H}) \mathbf{G}_a^-, \end{cases} \quad (20)$$

where matrix $\mathbf{K}$ is the Kalman filtering gain and $\mathbf{I}$ represents unit matrix

Thus, the Kalman filtering prediction algorithm starts with an initial predicted state $\hat{\mathbf{s}}_0^-$ and a certain error covariance $\mathbf{G}_0^- = \mathbf{I}\mathbf{I}_0$ then executes the time update and measurement update by iterations until satisfying a certain termination condition. Finally, we can get the predicted trajectory of each mobile user, including the position and the velocity information at each time slot.

4.2. Sociality Association. In this subsection, we consider the influence of sociality association on choosing appropriate edge servers for offloading tasks. Since there is an acquaintance effect, we mainly take the history connections into account. And we use the Renyi entropy to describe the selection preference. A higher entropy represents higher uncertainty on a specific selection. Specifically, the Renyi entropy is denoted as

$$H_\alpha^i(P) = \frac{1}{1-\alpha} \log \left(\sum_{j=1}^M p_{i,j}^\alpha \right), \quad (21)$$

where α is a general parameter in Renyi entropy. Different values of α represent different entropy; for example, when $\alpha = 1$, it is the Shannon entropy as $\lim_{\alpha \rightarrow 1} H_\alpha(P) = \sum_{j=1}^M \log(1/p_{i,j})$. And $p_{i,j}$ is the probability of U_i communicates with S_j . A smaller value of $H_\alpha^i(P)$ means a higher possibility that U_i will offload its tasks to the server j_m with the maximum value of $j_m = \arg_j \max p_{i,j}, \forall j \in \{1, 2, \dots, M\}$.

Let $\mathcal{V}$ represent the set of social association matrix between edge servers and mobile users. $\mathcal{V}_{i,j,a} = \langle i, j, t_a \rangle$ represents that U_i communicates with S_j during $t = aT$. For example, $\mathcal{V}_{i,\cdot} = (\langle t_3 \rangle, \langle t_1, t_{15} \rangle, 0, \dots, \langle t_{10}, t_{12}, t_{19} \rangle)$ is the association vector for U_i , demonstrating that U_i has con-

nected with S_1 at t_3 , S_2 at t_1 and t_{15} , and S_M at t_{10}, t_{12}, t_{19} and never connected with S_3 . The set of request user i was proposed for edge server j . Let $w_{i,j}$ denote the communication frequency between U_i and S_j , e.g., $w_{i,1} = 1, w_{i,2} = 2, w_{i,3} = 0, \dots, w_{i,M} = 3$. Define $\Omega_i = \sum_j w_{i,j}$ representing the total number of U_i communicate with all edge servers. Thus, the probability $p_{i,j}$ is calculated as

$$p_{i,j} = \frac{w_{i,j}}{\Omega_i}, \quad \text{for } U_i \in \mathcal{U}, S_j \in \mathcal{S}. \quad (22)$$

Similarly, $\mathcal{V}_{\cdot,j}$ can be easily obtained, denoting the social association between S_j and all users and $q_{j,i}$ can be also calculated as

$$q_{j,i} = \frac{w'_{j,i}}{\Omega'_j}, \quad \text{for } U_i \in \mathcal{U}, S_j \in \mathcal{S}. \quad (23)$$

4.3. AP Clustering. To accelerate the determination on connection with appropriate edge servers, we propose mobility-aware and sociality-associate clustering algorithm. Without the need of giving the number of clusters, the affinity propagation- (AP-) based clustering is employed. For the sake of striking the balance between accuracy and complexity, the update window is set as L slots.

4.3.1. Similarity Definition. The AP algorithm takes similarity, $s(i, j)$, as a measure describing the resemblance between nodes. $s(i, j) \geq s(i, j')$ when node U_i has higher similarity with S_j than with $S_{j'}$. In the mobility model, the similarity is probable to be higher when two nodes have closer distance between them. Thus, we define the similarity as the reciprocal of the average distance between nodes with future L slots according to the mobility prediction results.

$$\begin{aligned} d(i, j) &= \frac{\sum_{t=aT}^{t=[a+(L-1)]T} |(\hat{p}_x, \hat{p}_y)_t^i - (p_x, p_y)_t^j|}{L}, \\ d(i, i') &= \frac{\sum_{t=aT}^{t=[a+(L-1)]T} |(\hat{p}_x, \hat{p}_y)_t^i - (\hat{p}_x, \hat{p}_y)_t^{i'}|}{L}, \\ d(j, j') &= \frac{\sum_{t=aT}^{t=[a+(L-1)]T} |(p_x, p_y)_t^j - (p_x, p_y)_t^{j'}|}{L}. \end{aligned} \quad (24)$$

Thus, we have

$$\begin{aligned} s(i, j) &= \frac{1}{d(i, j)}, \\ s(i, i') &= \frac{1}{d(i, i')}, \\ s(j, j') &= \frac{1}{d(j, j')}. \end{aligned} \quad (25)$$

```

Input:
•The user set  $\mathcal{U}$  and edge servers set  $\mathcal{S}$ ;
•The positions of edge servers  $(p_x, p_y)^j$ ;
•History connection relationship  $\mathcal{V}$  between  $U_i$  and  $S_j$ ;
Output:
• $\mathcal{F}$ : the partition of all nodes into clusters;
• $\mathcal{CH}$ : the set of cluster centers;
1: function KALMANFILTERING $\mathcal{U}$ .
2:   Initialize  $\mathbf{s}_0$  and  $\mathcal{P}_0$ ;
3:   for Each  $a \in [1, L-1]$  do
4:     Time Update according to (19);
5:     Measurement Update according to (20);
6:   end for
7:   Output mobility prediction results:
8:    $\mathbf{s}^i = \{\mathbf{s}_1, \mathbf{s}_2, \dots, \mathbf{s}_{L-1}^i\}$ .
9: end function
10: function PROBABILITYCOMPUTATION $\mathcal{V}$ 
11:   Build matrix  $V_{i,j}$  and  $V_{j,i}$ ;
12:   Calculate  $w_{i,j}$ ,  $\Omega_i$ , then compute  $p(i, j)$  according to (22);
13:   Calculate  $w'_{j,i}$ ,  $\Omega'_j$ , then compute  $q(j, i)$  according to (23);
14:   Output  $p(i, j)$  and  $q(j, i)$ ,  $\forall U_i \in \mathcal{U}, S_j \in \mathcal{S}$ .
15: end
16: function APCLUSTERING $(p_x, p_y)^j, \mathbf{s}^i, p(i, j), q(j, i)$ 
17:   Calculate the distances  $d(n, m)$  based on (24);
18:   Calculate the similarities  $s(n, m)$  based on (25);
19:   Initialize  $a(n, m)$  to be zero;
20:   Initialize  $r(n, m)$  according to the type of nodes;
21:   Update the responsibility  $r(n, m)$  based on (26);
22:   Update the availability  $a(n, m)$  based on (27);
23:    $\mathcal{CH} = \{CH \mid CH = \arg_m \max [a(n, m) + r(n, m)]\}$ ;
24:   Calculate  $\mathcal{F}_{CH_n} = \{m \mid CH_m = CH_n\}$ ;
25:    $\mathcal{F} = \{\mathcal{F}_{CH_n} \mid CH_n \in \mathcal{CH}\}$ ;
26:   Output  $\mathcal{CH}$  and  $\mathcal{F}$ 
27: end function

```

ALGORITHM 1: AP-based mobility-aware and sociality-associate clustering algorithm (AP-MASACA).

The smaller the distance is, the bigger the similarity and then, the higher the possibility that U_i and S_j belong to a same cluster. Assume the distance between any two different nodes is larger than 1; thus, the maximum value of similarity is set 1 such that $s(i, i) = s(j, j) = 1$. Note that i, i' refer to the subscripts of mobile users and j, j' are the subscripts of edge servers.

4.3.2. AP-Based Clustering Mobility-Aware and Sociality-Associate Clustering Algorithm. During the clustering, we generally use subscripts n, m to denote all nodes in the system. After similarity measures, there are other two important messages transferred between nodes [28, 29]: (i) responsibility, $r(n, m)$, is sent from node n to candidate exemplar node m , indicating how well the node m is suitable as the exemplar of node n ; (ii) availability, $a(n, m)$, is sent from candidate exemplar node m to node n , reflecting how appropriate it would be for the node n choose node m as its exemplar. The initial values are given according to the social association, i.e., for the case of $n = i$ and $m = j$; calculate $r(i, j) = q(j, i)$

and $a(i, j) = p(i, j)$; otherwise, the initial value is $a(n, m) = 0$, e.g., $a(i, i) = a(i, i') = a(j, j') = a(j, j) = 0$.

The update formulas for responsibility and availability are stated as

$$r(n, m) \leftarrow s(n, m) - \max_{m' \neq m} a(n, m') + s(n, m'), \quad (26)$$

$$a(n, m) \leftarrow \min_{n \neq m} \left\{ 0, r(n, m) + \sum_{n' \notin \{n, m\}} \max \{0, r(n', m)\} \right\}, \quad (27)$$

$$a(m, m) \leftarrow \sum_{n' \neq m} \max \{0, r(n', m)\}. \quad (28)$$

In order to avoid numerical oscillations, the AP algorithm introduces a convergence coefficient λ when updating information. Each piece of information is set to λ times its previous iteration updating value plus $1 - \lambda$ times the current information updating value. Among that, the attenuation

TABLE 2: Parameter settings.

Parameters	Values
The initial state vector in Kalman filter $\mathbf{s}^{(0)}$	$(0, 0, 0, \nu)^T$
The number of tasks to be executed	[10, 50]
The processing capacity of MEC servers	10
The transmission power P_{Tx}	-10 (dBm)
The standard deviation of shadowing σ	8 (dB)
The desired SNR γ_{th}	10 (dB)
The AWGN N_0	-95 (dBm)

coefficient λ is a number between 0 and 1, while in our simulation $\lambda = 0.5$ is used.

$$r_{t+1}(n, m) \leftarrow (1 - \lambda)r_{t+1}(n, m) + \lambda r_t(n, m), \quad (29)$$

$$a_{t+1}(n, m) \leftarrow (1 - \lambda)a_{t+1}(n, m) + \lambda a_t(n, m). \quad (30)$$

Availabilities and responsibilities are combined to select exemplar. The maximum value of $a(n, m) + r(n, m)$ identifies the exemplar. Upon convergence, each node's cluster head CH is defined by

$$CH = \arg \max_m \{r(n, m) + a(n, m)\} \quad (31)$$

A detailed AP-based clustering algorithm is shown in Algorithm 1.

5. QoS-Aware Offloading Scheme

According to the AP clustering results, each MU has a high probability to offload its tasks to the edge servers within a same cluster. During designing the clustering algorithm, we have considered the influence of history connections and the mobility prediction. To realize the objective given in P1, we take two more factors into account, encompassing the service ratio and the channel conditions between MU and edge servers.

Although it is overloaded to make edge servers execute all of computation tasks, edge servers are expected to serve as many mobile users as possible. In addition in most cases, users prefer to communicate with edge servers with more available resources rather than the ones that are trapped in resource competition. So as the available resource of edge servers decreases, their corresponding social strength may be affected to decrease. To visually evaluate the quality of computation offloading based on the AP-MASACA, we defined a QoS parameter average service ratio, which follows

$$\text{Sev}(i) = \frac{n_{\text{sev}_i}}{n_d}, i = 1, 2, \dots, k, \quad (32)$$

$$\overline{\text{Sev}} = \frac{\sum \text{Sev}(i)}{k}, \quad (33)$$

where $\text{Sev}(i)$ represents the service ratio of i -th edge server, n_{sev_i} represents the actual number of mobile devices attach-

FIGURE 4: The average offloading time delay.

FIGURE 5: The service ratio with different user scales.

ing the i -th edge server (assuming that every mobile user has only one device), and n_d represents the actual number of mobile devices served by any edge server.

By applying MASACA strategies, the system cost of computation offloading could be decreased. When signal propagates through wireless channels, it undergoes deleterious effects mainly characterized by path loss, multipath fading, and shadowing, while the computation links between mobile users and edge servers could be interrupted. The SINR outage probability is an important channel quality measure of communication links operating over composite fading/shadowing channels, which is defined as the probability that the received SNR γ drops below a certain threshold γ_{th} for a certain average SNR $\bar{\gamma}$ [30]:

$$P_{\text{out}}(\bar{\gamma}, r_{th}) = E[\theta(\gamma_{th} - \gamma)]. \quad (34)$$

Note that $P_{\text{out}}(\bar{\gamma}, r_{th})$ requires the SNR distribution. Considering that the SNR could be affected by the distance of links, SNR $\gamma(d)$ could be obtained following

$$P_{R_x}(d) = P_{Tx} - L(d) + W, \quad (35)$$

$$\gamma(d) = P_{R_x}(d) - N_0, \quad (36)$$

FIGURE 6: The connection ratio vs. number of tasks in single user scenario.

where P_{T_x} represents the transmission power, $L(d)$ represents the path loss, and W represents the shadowing gain factor. And N_0 represents the noise power of additive white Gaussian noise (AWGN) in a wireless communication environment.

Furthermore, according to the composite lognormal/gamma PDF introduced by [31], we could get the final expression about outage probability as follows:

$$f_{\sigma}(\sigma) = \frac{1}{\sqrt{2\pi}\zeta\alpha} \exp\left(-\frac{(\ln(\sigma) - \mu)^2}{2\zeta^2}\right), \quad (37)$$

$$P_{\text{out}}(d) = \frac{1}{2} \left\{ 1 + \operatorname{erf} \frac{\gamma - (P_{R_x} - L(d) - N_0)}{\sqrt{2}\sigma^2} \right\}. \quad (38)$$

We define the connection ratio to visually describe the performance for maintaining the sustainability of connections with edge servers in different cases:

$$P_{\text{con}}(d) = 1 - P_{\text{out}}(d). \quad (39)$$

6. Experiments and Result Analysis

In this section, we conduct simulation experiments to verify the efficiency of our proposed offloading scheme by comparing with an existing method employed in [32, 33] where mobile users randomly offload computation tasks to edge servers they have searched.

In our simulation, MEC servers are uniformly distributed within an area with side length of 200 m. The simulation settings are listed in Table 2. Under different numbers of MEC servers N and different moving speed of mobile users v , we conduct experiments to show the performance in terms of the time delay, the service ratio, and the connection ratio. The results are averaged over 1000 independent runs.

Figure 4 shows the influence of the number of mobile users on the average time delay. We can see that the average offloading time delay increases with the expansion of mobile users' scale for each method. This is an intuitive result because the computation capability of MEC servers is limited and more users need to wait for a specific period before being serviced. For the same number of mobile users, our proposed scheme obtains a lower average delay, outperforming the original random offloading method. This demonstrates the effectiveness of our proposed scheme.

Figure 5 shows that the number of mobile users serviced by edge servers will increase with the expansion of mobile users' scales. Compared with the case that mobile users offload computation tasks randomly, we can see the improvements that applying AP-MASACA could bring more chances for mobile users to be served. At the same time, it can be seen that the system performance was not greatly stable as the number of mobile users increases, which means that there may be available resources of some edge servers not fully utilized and the offloading strategy can be further optimized.

Figure 6 shows the connection ratio under different velocities in the single-user scenario. We can see that the connection ratio of each method declines with the increase of the number of computation tasks. This is because the competition for resources becomes more intense with the increase of task amount. Meanwhile, the decline rate of our proposed scheme is slower than that of the original random case. For the same number of computation tasks, our proposed scheme has a higher connection ratio than the original random method. This demonstrates our proposed scheme can improve the sustainability of connections between MU and MEC servers. Moreover, it is obvious that the larger the number of MEC server, N , is, the higher the connection ratio. By comparing the results shown in Figures 6(a) and 6(b), we can know that the connection ratio decreases with the increasing speed. This is because the distance between

FIGURE 7: The connection ratio vs. number of tasks in multiuser scenario.

the MU and the selected MEC servers changes faster with the increasing speed.

Figure 7 shows the connection ratio of multiuser scenario. As the number of MEC servers increases, the system performance is getting better. And it shows that the performance in the multiuser scenario is better than that in the single-user scenario. Figure 6 shows the connection ratio of multiuser when the moving speed is 80 km/h. From the simulation results, it has been shown that the connection ratio is improved by applying the proposed method especially when mobile users move at a high speed.

7. Conclusion

In this paper, we design a task offloading strategy by considering the mobility, the sociality, and QoS factors. First, we employ the Kalman filtering algorithm to do mobility prediction of mobile users' trajectory; then, we compute the sociality association parameter between mobile users and edge servers based on the history connection relationship. By applying the mobility prediction results and the social association parameter as input parameters of affinity propagation algorithm, we propose the mobility-aware and sociality-associate clustering algorithm (MASACA). By applying the clustering results as candidate edge server set, we devise QoS utility function of a given edge server based on the available service ratio and the quality of channel link. At last, based on the designed QoS utility function, we select the candidate edge server with the highest QoS utility as the target edge server to offload tasks. Numerical simulation results demonstrate that our offloading strategy can enhance the data processing capability of power-constrained networks and cut down the computation delay.

Data Availability

In the simulation section, we use the social network data "usense" provided at <http://www.crawdad.org/copelabs/usense/20170127/index.html>. In the data set, we employ the

distances between users, the encounter durations, and other related data as the input parameters of our proposed clustering algorithm.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the Fundamental Research Funds for the Central Universities under Grant 2019JBZ001 and the Key Project of the National Natural Science Foundation of China under Grant 61931001.

References

- [1] Z. Cai and X. Zheng, "A private and efficient mechanism for data uploading in smart cyber-physical systems," *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 2, pp. 766–775, 2020.
- [2] Y. Huo, C. Meng, R. Li, and T. Jing, "An overview of privacy preserving schemes for industrial internet of things," *China Communications*, vol. 17, no. 10, pp. 1–18, 2020.
- [3] X. Zheng and Z. Cai, "Privacy-preserved data sharing towards multiple parties in industrial IoTs," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 5, pp. 968–979, 2020.
- [4] M. Patel, B. Naughton, and C. Chan, "Mobile-edge computing introductory technical white paper," *Mobile-Edge Computing (MEC) Industry Initiative*, pp. 1089–7801, 2014.
- [5] F. Vhora and J. Gandhi, "A comprehensive survey on mobile edge computing: challenges, tools, applications," in *2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC)*, pp. 49–55, Erode, India, March 2020.
- [6] Z. He, Z. Cai, S. Cheng, and X. Wang, "Approximate aggregation for tracking quantiles and range countings in wireless sensor networks," *Theoretical Computer Science*, vol. 607, no. 3, pp. 381–390, 2015.

- [7] Z. Cai, X. Zheng, and J. Yu, "A differential-private framework for urban traffic flows estimation via taxi companies," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 12, pp. 6492–6499, 2019.
- [8] Y. Ding, C. Liu, K. Li, Z. Tang, and K. Li, "Task offloading and service migration strategies for user equipments with mobility consideration in mobile edge computing," in *2019 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BD-Cloud/SocialCom/SustainCom)*, pp. 176–183, Xiamen, China, December 2019.
- [9] Y. Zhang, D. Niyato, and P. Wang, "Offloading in mobile cloudlet systems with intermittent connectivity," *IEEE Transactions on Mobile Computing*, vol. 14, no. 12, pp. 2516–2529, 2015.
- [10] N. Ahmed and B. K. Bhargava, "Towards dynamic QoS monitoring in service oriented architectures," in *Proceedings of the 5th International Conference on Cloud Computing and Services Science - CLOSER*, pp. 163–171, Lisbon, Portugal, 2015.
- [11] S. Guo, B. Xiao, Y. Yang, and Y. Yang, "Energy-efficient dynamic offloading and resource scheduling in mobile cloud computing," in *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*, pp. 1–9, San Francisco, CA, USA, April 2016.
- [12] C. A. Kamienski, F. F. Borelli, G. O. Biondi, I. Pinheiro, I. D. Zyrianoff, and M. Jentsch, "Context design and tracking for IoT-based energy management in smart cities," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 687–695, 2018.
- [13] Z. He, Z. Cai, J. Yu, X. Wang, Y. Sun, and Y. Li, "Cost-efficient strategies for restraining rumor spreading in mobile social networks," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 3, pp. 2789–2800, 2017.
- [14] J. Li, S. Cheng, Y. Li, and Z. Cai, "Approximate holistic aggregation in wireless sensor networks," in *2015 IEEE 35th International Conference on Distributed Computing Systems*, pp. 740–741, Columbus, OH, USA, June 2015.
- [15] M. Mehrabi, H. Salah, and F. H. P. Fitzek, "A survey on mobility management for MEC-enabled systems," in *2019 IEEE 2nd 5G World Forum (5GWF)*, pp. 259–263, Dresden, Germany, September 2019.
- [16] S. Wang, J. Xu, N. Zhang, and Y. Liu, "A survey on service migration in mobile edge computing," *IEEE Access*, vol. 6, pp. 23511–23528, 2018.
- [17] C.-L. Wu, T.-C. Chiu, C.-Y. Wang, and A.-C. Pang, "Mobility-aware deep reinforcement learning with glimpse mobility prediction in edge computing," in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, pp. 1–7, Dublin, Ireland, June 2020.
- [18] J. Xu, Y. Sun, L. Chen, and S. Zhou, "E2M2: energy efficient mobility management in dense small cells with mobile edge computing," in *2017 IEEE International Conference on Communications (ICC)*, pp. 1–6, Paris, France, May 2017.
- [19] Z. Cai, Z. He, X. Guan, and Y. Li, "Collective data-sanitization for preventing sensitive information inference attacks in social networks," *IEEE Transactions on Dependable and Secure Computing*, vol. 15, no. 4, pp. 577–590, 2018.
- [20] C.-M. Huang, C.-H. Shao, S.-z. Xu, and H. Zhou, "The social internet of thing (S-IOT)-based mobile group handoff architecture and schemes for proximity service," *IEEE Transactions on Emerging Topics in Computing*, vol. 5, no. 3, pp. 425–437, 2017.
- [21] Z. Cai and Z. He, "Trading private range counting over big IoT data," in *2019 IEEE 39th International Conference on Distributed Computing Systems (ICDCS)*, pp. 144–153, Dallas, TX, USA, July 2019.
- [22] C.-H. Wang, J. J. Kuo, D. N. Yang, and W. T. Chen, "Collaborative social Internet of Things in mobile edge networks," *IEEE Internet of Things Journal*, vol. 7, no. 12, pp. 11473–11491, 2020.
- [23] H. Zhou, L. Xu, and J. Zhang, "A physical-social-based group utility maximization framework for cooperative caching in mobile networks," in *2018 10th International Conference on Wireless Communications and Signal Processing (WCSP)*, pp. 1–7, Hangzhou, China, October 2018.
- [24] D. Belli, S. Chessa, L. Foschini, and M. Girolami, "A social-based approach to mobile edge computing," in *2018 IEEE Symposium on Computers and Communications (ISCC)*, pp. 00 292–00 297, Natal, Brazil, June 2018.
- [25] S. Cheng, Z. Cai, J. Li, and H. Gao, "Extracting kernel dataset from big sensory data in wireless sensor networks," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 4, pp. 813–827, 2017.
- [26] J. Jung, S. Chun, X. Jin, and K.-H. Lee, "Quantitative computation of social strength in social internet of things," *IEEE Internet of Things Journal*, vol. 5, no. 5, pp. 4066–4075, 2018.
- [27] Y. Wang, M. Ding, Z. Chen, and L. Luo, "Caching placement with recommendation systems for cache-enabled mobile social networks," *IEEE Communications Letters*, vol. 21, no. 10, pp. 2266–2269, 2017.
- [28] Q. Gao, Y. Wang, X. Cheng, J. Yu, X. Chen, and T. Jing, "Identification of vulnerable lines in smart grid systems based on affinity propagation clustering," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 5163–5171, 2019.
- [29] B. J. Frey and D. Dueck, "Clustering by passing messages between data points," *Science*, vol. 315, no. 12, pp. 927–976, 2007.
- [30] F. Yilmaz, "On the relationships between average channel capacity, average bit error rate, outage probability, and outage capacity over additive white gaussian noise channels," *IEEE Transactions on Communications*, vol. 68, no. 5, pp. 2763–2776, 2020.
- [31] M.-J. Ho and G. Stuber, "Co-channel interference of microcellular systems on shadowed Nakagami fading channels," in *IEEE 43rd Vehicular Technology Conference*, pp. 568–571, Secaucus, NJ, USA, 1993.
- [32] Y. Liu, S. Wang, J. Huang, and F. Yang, "A computation offloading algorithm based on game theory for vehicular edge networks," in *2018 IEEE International Conference on Communications (ICC)*, pp. 1–6, Kansas City, MO, USA, May 2018.
- [33] G. Fan, W. Bai, X. Gan, X. Wang, and J. Wang, "Social-aware content sharing in D2D communications: an optimal stopping approach," in *2018 IEEE International Conference on Communications (ICC)*, pp. 1–6, Kansas City, MO, USA, May 2018.