IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received January 31, 2020, accepted February 18, 2020, date of publication March 3, 2020, date of current version March 16, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2978093

Particle Swarm Based Service Migration Scheme in the Edge Computing Environment

LIANG LIANG[1], JINTAO XIAO[1], ZHI REN[2], ZHENGCHUAN CHEN[1], (Member, IEEE), AND YUNJIAN JIA[1]

[1]School of Microelectronics and Communication Engineering, Chongqing University, Chongqing 400044, China
[2]School of Communication and Information Engineering, Chongqing University of Posts and Telecommunications, Chongqing 400065, China

Corresponding author: Liang Liang (liangliang@cqu.edu.cn)

This work was supported in part by the China Postdoctoral Science Foundation under Grant 2016M602671, in part by the National Natural Science Foundation of China under Grant 61971077 and Grant 61901066, in part by the Project Supported by Chongqing Key Laboratory of Mobile Communications Technology under Grant cqupt-mct-201902, and in part by the Chongqing Science and Technology Commission under Grant cstc2019jcyjmsxmX0575.

ABSTRACT With the development of Mobile Edge Computing (MEC), it has become a key technology to realize the vision of the Internet of Things. In MEC, users can upload tasks to edge nodes for faster processing speed and lower local energy consumption. However, as the mobility of users and the limited resources of the edge nodes, some edge nodes cannot provide high-quality services. In this case, we study service migration strategy in the MEC system to migrate services from the initial nodes to other edge nodes that can provide services to meet the needs of users. By making service migration decision and allocating computation resource, our work minimizes the delay and the energy consumption caused by finishing tasks. Specifically, we set up an efficient service migration model and formulate the service migration problem as a non-linear 0-1 programming problem. To solve this problem, we design a Particle Swarm based Service Migration scheme (PSSM) which includes Queuing Delay Prediction algorithm (QDP), Delay-aware Computation Resource Allocation algorithm (DCRA), and Modified Quantum Particle Swarm algorithm (MQPS). For evaluating the performance of the proposed PSSM, we conduct simulation in a practical scenario. The results demonstrate that our scheme not only can effectively reduce delay and energy consumption, but also improve the processing capability of servers.

INDEX TERMS Mobile edge computing, service migration, mobility, particle swarm algorithm.

I. INTRODUCTION

Mobile Cloud Computing (MCC), as the integration of cloud computing and mobile computing, has been widely used in the past few years. Generally, the large centralized data center away from users. The data interaction between the user and the data center must be carried out through the radio access network, the backhaul network and the core network. In such case, excessive delay will be produced by communication control, routing and other operations. Moreover, the cloud will process a large number of tasks at the same time, and the computation resource assigned to a task is limited. Hence, the processing delay of a task cannot be ignored and congestion may occur among massive users. Besides, the rapid emergence of many new business applications such as augmented reality, virtual reality, and car networking have higher requirements for network transmission and data distribution computation. In order to deal with the above challenges, Mobile Edge Computing (MEC) has been proposed [1]. The edge in the MEC refers to the places that are closer to users than the data center. MEC scenario deploys a part of resource (such as storage resource and computing resource) to the edge of the network to provide service to users [2], [3], so that the data generated by users would not need to be processed in the data center. In this way, the congestion of the core network can be alleviated, and the response time of users can be reduced significantly.

With the rapid development of the Internet of Things (IoT), the number of users and data traffic are increasing dramatically, so MEC has become an indispensable key technology for realizing the vision of IoT. Meanwhile, as more and more mobile devices exist in the network, service migration has

The associate editor coordinating the review of this manuscript and approving it for publication was Luca Chiaraviglio.

gradually become an important problem [4]. In the MEC scenario, because of the limited coverage of edge servers, the mobility of user terminals (such as smartphones and smart vehicles [5]) may cause a significant degradation in the Quality of Service (QoS). Furthermore, due to the limited resource of the edge servers, changes in network status will bring greater impact on the quality of service. Particularly, when an edge server is overloaded, its users cannot get enough resource from it, and thus some services need to be migrated to adjacent servers that can fulfill the requirements. Since the service is packaged in a separate virtual machine to process, service migration can be considered as the virtual machine dynamic migration [6].

At present, there are a lot of researchers studying service migration in MEC related scenarios. In [7], the authors provided an overview of fog computing, and proposed a general architecture based on virtual machine migration in fog computing, and then discussed the benefits and challenges of virtual machine migration under this architecture. In [8], a general layered architecture was proposed for migrating services encapsulated in virtual machines. This work divided the migrated application into multiple layers and greatly reduced the amount of transferred data. Considering the costs and benefits of service migration, the work in [9] proposed an edge computing migration strategy based on multi-attribute decision to balance costs and benefits. The method considered various indicators as decision matrices to select the optimal migration target, but it did not consider the impact of user mobility. In [10], the authors used the Markov Decision Process (MDP) to model the service migration process to formulate a decision strategy to determine whether to migrate services when the relevant user equipment was at a distance from the source data center. In [11], the authors designed an edge cloud migration decision system, which solved the problem of when the user migrated and where to migrate. Authors in [12] used the MDP framework to formulate a sequential decision problem for service migration. Although this work considered the trade-off between network overhead and the distance between the user and the edge cloud, the resource constraints and delay requirements of the edge cloud was not considered. In [13], the authors considered a one-dimensional asymmetric random walk model, and modeled the service migration problem as a MDP. However, the authors only considered the cost of migration, without considering QoS.

Actually, users' mobility and network condition fluctuation may cause great degradation of QoS. In order to provide better services to users, this paper designs a service migration model and a Particle Swarm based Service Migration scheme (PSSM). PSSM includes three algorithms, which are Queuing Delay Prediction algorithm (QDP), Delay-aware Computation Resource Allocation algorithm (DCRA), and Modified Quantum Particle Swarm algorithm (MQPS). Specifically, in the service migration model, we use the random walk model to simulate the motion status of users. Moreover, we assume the user status, network status and edge server status are collected in real time, so that the migration model can effectively reflect the real situation. On this basis, we design QDP to predict the queuing delay of tasks and design DCRA to allocate computation resource dynamically to improve the efficiency of servers. Next, we formulate the service migration problem as a non-linear 0-1 programming problem. By taking advantages of quantum particle swarm optimization, we design MQPS to solve this problem.

The main contributions of this paper are as follows:

1) We design a queuing delay prediction algorithm to predict the queuing delay of tasks. This algorithm not only can predict the average queue delay, but also can predict the queuing delay for individual task.
2) We design a delay-aware computation resource allocation algorithm to dynamically allocate computation resource according to the needs of the task and the status of edge servers. This algorithm can not only meet QoS requirement of tasks but also improve the efficiency of servers.
3) We design a modified quantum particle swarm algorithm to solve the service migration problem. This algorithm has better search efficiency and is easier to break away from local extreme point constraints compared to traditional particle swarm optimization.

The remainder of this paper is organized as follows. Section 2 presents system model and problem formulation. PSSM is presented in Section 3. Section 4 demonstrates the simulation results. Finally, Section 5 concludes this study.

II. SYSTEM MODEL AND PROBLEM FORMULATION

The service migration scenario shown in Fig. 1 includes a local network controller, a control base station, and some small base stations. The local network controller manages the mobility of users and has four modules, which are status collection module, delay prediction module, server selection module, and service migration module. The control base station is used to transmit control signal, and the small base station is mainly used to transmit data. In this scenario, servers are deployed in the small base stations and have

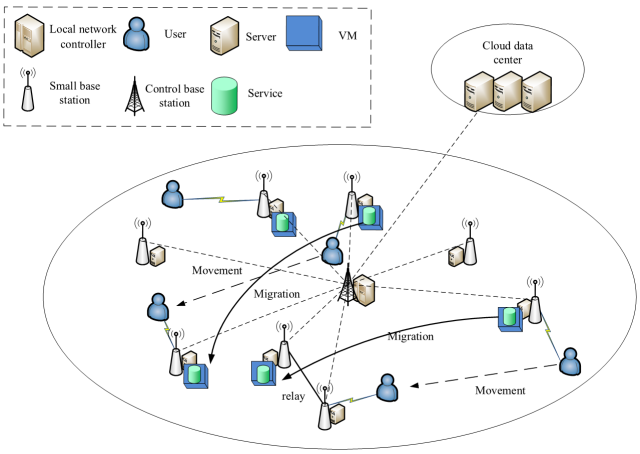


FIGURE 1. Service migration scenario.

some computing resources and storage resources. Assuming the storage resources are enough and the hot information stored in servers is downloaded periodically from the local network controller, we mainly consider computing intensive services in this paper. When the target base station cannot communicate with the user directly, it needs the help of a relay base station.

As mentioned above, the local network controller has four modules. The status collection module collects the user status, servers status and network status periodically. The delay prediction module predicts the delay of the tasks. The server selection module selects the target server, and the service migration module control the migration process. Whenever a user produces a task, the delay prediction module predicts the available computation resource and the response time according to the information collected by the status collection module. When the delay of the task fails to meet the requirement, the server selection module will select the server with the largest reward of delay and energy consumption, and then the service migration module migrates the virtual machine from the original server to the target server.

A. SEVER MODEL

In the server model, each server allocates a virtual machine to a service, and each service consists of multiple tasks generated by a user. After a task is processed, it is possible that the next task has not yet arrived. At this time, the virtual machine will be in the standby status until the next task comes to activate it. If all tasks in the service are finished, the virtual machine will be logged off. In our work, we denote $N_{server} = \{1, 2, \ldots, N_s\}$ and $N_{user} = \{1, 2, \ldots, N\}$ as a set of servers and a set of users, respectively.

The computation resource required by a task is positively related to the size of the task. If each virtual machine has a fixed computation resource, both processing delay and transmission delay of a task with a larger size will be longer, so that the total delay may not meet the requirement. Moreover, if the servers with light loads still allocate default amount of resource to each task, the efficiency of the servers will be seriously affected. Take the Ultra Reliable and Low Latency Communication (uRLLC) task as an example. This kind of tasks has a high delay requirement, thus it is essential to reduce the processing delay by allocating computation resource reasonably.

B. DELAY MODEL

The delay involved in this paper includes the random access delay, the data transmission delay, the queuing delay, the processing delay, the service migration delay, and the relay transmission delay. In particular, as the number of users that initiate random access in each time slot is very small and the probability of random access failure is very low, we set the random access delay of the tasks to be a fixed value.

The detailed definitions of different parts of delay are as follows.

1) TRANSMISSION DELAY

The data transmission delay is defined as the time taken for data transmission between a user and a base station. Considering the downlink speed is much faster than the uplink speed, we mainly consider the uplink transmission delay. It is related to the amount of data and the transmission speed, so the data transmission delay of the task from user i to base station j at time t is

$$d_{tr}^{i,t} = \frac{S_{task}^{i,t}}{V_{tr}^{i,j,t}}, \tag{1}$$

where $S_{task}^{i,t}$ is the task size of user i, $V_{tr}^{i,j,t}$ is the data transmission speed and can be expressed as

$$V_{tr}^{i,j,t} = B\log_2(1 + \frac{P_{tx}^{i} H^{i,j,t}}{N_{noise}}), \tag{2}$$

where B is the channel bandwidth, P_{tx}^{i} is the transmission power of user i, $H^{i,j,t}$ is the channel gain between user i and base station j, and N_{noise} is the noise power. Since the used channels are orthogonal, we only consider the background noise.

2) RELAY TRANSMISSION DELAY

The relay transmission delay is defined as the transmission time from the relay to the target base station. As some users cannot communicate with the target base station directly, they need other base stations as relays to assist transmission. Hence, if the task from user i needs to be transmitted from relay j to the target base station j', the relay transmission delay is

$$d_{relay}^{i,t} = \frac{S_{task}^{i,t}}{V_{tr}^{j,j',t}}. \tag{3}$$

3) PROCESSING DELAY

The processing delay is defined as the time of a task processed in the server, which is the ratio of the required computation resource to the computation speed provided by the server. So the processing delay is

$$d_{de}^{i,t} = \frac{N_{cpu}^{i,t}}{V_{cpu}^{i,t}}, \tag{4}$$

where $N_{cpu}^{i,t} = \varphi S_{task}^{i,t}$ represents the computation amount required by the task of user i at time t, $V_{cpu}^{i,t}$ is the computation speed for the task of user i at time t, and φ is a proportionality coefficient.

4) SERVICE MIGRATION DELAY

The service migration delay is defined as the transmission time of the VM from the original server to the target server. Because the tasks in the same service may need some processing results of other previous tasks and some related configuration information, all tasks of a service will be processed in the same VM. When the server cannot meet the minimum QoS requirement of the task, the VM located in

this server needs to be migrated to another one. Hence, if the service for user i needs to migrate from original server j to the target server j', the service migration delay is

$$d_{mi}^{i,t} = \frac{S_{vm}^{i,t}}{V_{tr}^{j,j',t}}, \tag{5}$$

where $S_{vm}^{i,t}$ is the size of the virtual machine. As the size of the virtual machine is proportional to the amount of the data that have been processed in the service, so we have

$$S_{vm}^{i,t} = \eta \sum S_{task}^{i,t}, \tag{6}$$

where η is a proportionality coefficient.

5) QUEUING DELAY

The queuing delay is defined as the waiting time of the task from reaching server moment to starting processing moment. If the server has heavy load, the task needs to queue in the reaching sequence. When the previous tasks have been processed, a corresponding VM will be activated for the task in the queue.

C. ENERGY CONSUMPTION MODEL

The energy consumption considered in this paper includes transmission energy consumption, relay energy consumption, processing energy consumption, and migration energy consumption.

1) TRANSMISSION ENERGY CONSUMPTION

Transmission energy consumption includes energy consumption for sending and receiving data. Assuming that the transmitting power of the user is proportional to the receiving power of the base station, the transmission energy consumption of the task of user i at time t is

$$E_{tr}^{i,t} = (1+\phi)\, P_{tx}^{i} d_{tr}^{i,t}, \tag{7}$$

where ϕ is the ratio of transmitting power to receiving power.

2) RELAY ENERGY CONSUMPTION

Relay energy consumption includes energy consumption for sending data at the relay and receiving data at the target base station. The relay energy consumption of the task is

$$E_{relay}^{i,t} = (1+\phi)\, P_{bs} d_{relay}^{i,t}, \tag{8}$$

where P_{bs} is the transmitting power of base station.

3) PROCESSING ENERGY CONSUMPTION

When the server needs to allocate a virtual machine to process the task of user i, the processing energy consumption is

$$E_{cpu}^{i,t} = P_{cpu}^{i,t} d_{de}^{i,t}, \tag{9}$$

where $P_{cpu}^{i,t}$ is the power of the virtual machine allocated to the task and it is defined as

$$P_{cpu}^{i,t} = \frac{V_{cpu}^{i,t}}{V_{server}} \left(P_{\max} - P_{idle}\right), \tag{10}$$

where $P_{\max}$ is the power of the server at full load, P_{idle} is the power of the server during idle time, and V_{server} is the total computation resource of the server.

4) SERVICE MIGRATION ENERGY CONSUMPTION

Service migration energy consumption includes energy consumption for transmitting VM from the original base station to the target base station. It can be described as

$$E_{mi}^{i,t} = (1+\phi)\, P_{bs} d_{mi}^{i,t}. \tag{11}$$

D. PROBLEM FORMULATION

Our work needs to decide when the service migration is needed and where the service is migrated to. In order to avoid frequent migration, a delay threshold is set and the service migration is triggered only when the delay of the task exceeds the threshold. In addition, the choice of migration target needs to maximize the benefits brought by the service migration. We define two variables $S_i, X_i \in N_{server}$ to indicate the target base station and the relay base station selected by user i respectively. If $X_i = S_i$, the user does not need a relay. Define a binary variable J_i as

$$J_i = \begin{cases} 1, & S_i \neq X_i \\ 0, & S_i = X_i. \end{cases} \tag{12}$$

Then the total delay of the task from user i at time t is

$$T^{i,t} = d_{tr}^{i,t} + d_{qu}^{i,t} + d_{de}^{i,t} + d_{mi}^{i,t} + J_i d_{relay}^{i,t}, \tag{13}$$

and the total energy consumption is

$$E^{i,t} = E_{tr}^{i,t} + E_{cpu}^{i,t} + E_{mi}^{i,t} + J_i E_{relay}^{i,t}. \tag{14}$$

$T_1^{i,t}$ and $T_2^{i,t}$ indicate the delay before and after the service migration of user i, respectively. $E_1^{i,t}$ and $E_2^{i,t}$ indicate the energy consumption before and after the service migration, respectively. In order to compare at the same level, we normalize delay and energy consumption as

$$\begin{aligned} T_1^{*,i,t} &= \frac{T_1^{i,t}}{\sum_{i=1}^{N}\left(T_1^{i,t}+T_2^{i,t}\right)}, \quad T_2^{*,i,t} = \frac{T_2^{i,t}}{\sum_{i=1}^{N}\left(T_1^{i,t}+T_2^{i,t}\right)}, \\ E_1^{*,i,t} &= \frac{E_1^{i,t}}{\sum_{i=1}^{N}\left(E_1^{i,t}+E_2^{i,t}\right)}, \quad E_2^{*,i,t} = \frac{E_2^{i,t}}{\sum_{i=1}^{N}\left(E_1^{i,t}+E_2^{i,t}\right)}. \end{aligned} \tag{15}$$

The benefit obtained from migration is

$$A = \sum_{i=1}^{N}\left(\beta_1\left(T_1^{*,i,t} - T_2^{*,i,t}\right) + \beta_2\left(E_1^{*,i,t} - E_2^{*,i,t}\right)\right). \tag{16}$$

We define the migration decision problem as an optimization problem:

$$\max\ A \tag{17}$$

$$\text{subject to: } S_i, X_i \in N_{server}, \quad i \in I_m, \tag{17a}$$

$$T_2^{i,t} \le D_e, \quad i \in I_e, \tag{17b}$$
$$T_2^{i,t} \le D_u, \quad i \in I_u, \tag{17c}$$
$$P_{tx}^{i} H^{i,X_i,t} \ge P_{rx\min}, \quad i \in I_m, \tag{17d}$$
$$P_{out}^{i,X_i,t} \le P_{out\max}, \quad i \in I_m. \tag{17e}$$

where I_m denotes a set of users that need service migration. I_e and I_u denote a set of users that generate eMBB tasks and uRLLC tasks respectively. D_e and D_u represent the delay threshold of eMBB tasks and uRLLC tasks respectively. $P_{out}^{i,j,t}$ represents the interruption probability of the transmission between user i and base station j at time t. $P_{rx\min}$ and $P_{out\max}$ denote the signal reception threshold and the interruption probability threshold respectively.

III. PARTICLE SWARM BASED SERVICE MIGRATION SCHEME

To solve the problem (17), we propose a particle swarm based service migration scheme (PSSM). In PSSM, we design QDP to estimate the queuing delay for each task. Then, DCRA is designed to allocate computation resource dynamically. Based on the delay information and computation resource of servers, we select the candidate server for service migration. Finally, we design MQPS to realize service migration with the maximum benefits.

A. QUEUING DELAY PREDICTION ALGORITHM

Generally, tasks generated by users follow Poisson flow. However, the time that a task reaches the server is affected by the transmission delay and the server selection. Therefore, the arrival of tasks does not obey the Poisson distribution, and we cannot directly use the queuing theory model to calculate the queuing delay. In this case, we propose QDP to predict the queuing delay in Algorithm 1. QDP can estimate the task queuing delay based on the remaining processing time of being processed tasks and the required processing time of waiting tasks. In Algorithm 1, we denote td_m as the remaining processing time of a task and denote d_{de}^n as the required processing time of a task.

B. DELAY-AWARE COMPUTATION RESOURCE ALLOCATION ALGORITHM

Based on the predicted queuing delay of each task, we design DCRA to allocate the available processing capabilities dynamically. According to the delay threshold, we first calculate the minimum computation resource V_{need} for each task. In some cases, the current remaining computation resource of the server may be less than V_{need}. If we wait for the server to idle, the processing delay needed to be shorten to meet the delay threshold, and then V_{need} will become larger. This situation seriously affects the performance of subsequent tasks and the QoS. In view of this, we set a minimum computation resource threshold $V_{u\min}$ for the uRLLC task, and the capability allocated to the uRLLC task is expressed as

Algorithm 1 Queuing Delay Prediction Algorithm

Input: $td_1, td_2, \ldots, td_m, d_{de}^1, d_{de}^2, \ldots, d_{de}^n$
Output: $d_{qu}^{i,t}$

```
Set b = [td_1, td_2, ..., td_m], p = 0, a = 1, and q = 1
for T = 1 to 50 do
    Arrange the numbers in array b in ascending order
    for T1 = a to m do
        td_T1 = td_T1 − 1
        if td_T1 = 0 then
            p = p + 1, a = a + 1 and update V_remain
        end if
    end for
    for c = 1 to 5 do
        if V_remain meets the requirements of the qth task
            in the queue then
            d_qu^q = b[p], td_{m+1} = d_qu^q + d_de^q, m = m + 1
            Let q = q + 1 and update V_remain
        else
            Break
        end if
    end for
    if q > n then
        Break
    end if
end for
Let d_qu^{i,t} = d_qu^n + d_de^n
```

$$V_u = \begin{cases} V_{u\min} + \dfrac{V_{remain} - V_{u\min}}{2}\left(1 + \dfrac{\sum_{\alpha=1}^{n} \chi_\alpha V_{free}^{\alpha}}{V_{remain} + \sum_{\alpha=1}^{n} \chi_\alpha V_{free}^{\alpha}}\right), \\ \qquad \text{if } V_{need} < V_{u\min} < V_{remain} \\ V_{need} + \dfrac{V_{remain} - V_{need}}{2}\left(1 + \dfrac{\sum_{\alpha=1}^{n} \chi_\alpha V_{free}^{\alpha}}{V_{remain} + \sum_{\alpha=1}^{n} \chi_\alpha V_{free}^{\alpha}}\right), \\ \qquad \text{if } V_{u\min} < V_{need} < V_{remain} \\ V_{u\min}, \qquad \text{if } V_{u\min} < V_{remain} < V_{need} \end{cases} \tag{18}$$

where V_{free}^{α} is the released computation resource in the subsequent αth time slot, and χ_α is the weight for the subsequent αth time slot. The larger α is, the smaller the weight is. To meet the minimum requirement of QoS, the required computation resource is

$$V_{need} = \frac{N_{cpu}^{i,t}}{D_u - D_{left}^{i,t}}, \tag{19}$$

where D_u is the delay threshold of the uRLLC task, and $D_{left}^{i,t}$ is total delay except for the processing delay of a task from user i.

For the Enhance Mobile Broadband (eMBB) task, the delay requirement is lower than the uRLLC task. So most of the time, the allocated computation resource will be less than that of the uRLLC task. When the server load is light, the computation resource allocated to the eMBB task will have an upper limit as

$$V_e = \begin{cases} V_{e\min}, & \text{if there are other tasks} \\ & \text{in the queue} \\ \min(V_{remain}, 2V_{e\min}), & \text{otherwise.} \end{cases} \quad (20)$$

C. CANDIDATE SERVER SELECTION

When we decide to migrate a service, we need to select a migration target among multiple candidate servers. Choosing a candidate server needs to consider the signal strength and the link stability.

The connectivity between user i and base station j changes as the user moves. The probability density function of the Signal to Noise Ratio (SNR) of the received signal obeys a lognormal distribution as

$$f(x) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left\{-\frac{\left(x - (P_{tx}^{i}H^{i,j,t} - N_{noise})\right)^2}{2\sigma^2}\right\}, \quad (21)$$

where σ is the standard deviation of lognormal shadowing gain.

When the SNR of the received signal is less than the threshold Γ, the communication will be interrupted, and the interruption probability equals to the probability of the signal SNR less than Γ [14]. Therefore, the interruption probability between user i and base station j is defined as

$$P_{out}^{i,j,t} = \frac{1}{2}\left\{1 + erf\left(\frac{-(P_{tx}^{i}H^{i,j,t} - N_{noise})}{\sqrt{2\sigma^2}}\right)\right\}. \quad (22)$$

The candidate server j for device i needs to meet the following conditions:

$$P_{rx}^{i,j,t} \geq P_{rx\min}, \qquad P_{out}^{i,j,t} \leq P_{out\max}. \quad (23)$$

D. MODIFIED QUANTUM PARTICLE SWARM ALGORITHM

The problem (17) can be seen as a nonlinear 0-1 programming problem, which is an NP-complete problem. As the optimal solution cannot be directly obtained, we propose MQPS to solve it in Algorithm 2.

The particle swarm optimization [15] is a swarm-based stochastic algorithm. It initializes the random location of each individual in the swarm, then iterates according to the collaboration and shared information until the optimal solution is found. In MQPS, a particle denotes a solution of the problem and a component of a particle denotes the server selection of a migrated user. When the number of iterations increases, the search speed and accuracy of this algorithm will change more reasonably.

The specific steps of MQPS are as follows:

Step 1: generate primary particle swarm. Since the Logistic chaotic map has the characteristics of ergodicity, regularity, randomness and sensitivity to initial values, we use the following Logistic chaotic map to initialize the particle swarm:

$$S_{i,n,1} = \lceil s_{i,n}N_s \rceil, \quad (24)$$
$$X_{i,n,1} = \lceil x_{i,n}N_s \rceil, \quad (25)$$
$$s_{i,n+1} = 4s_{i,n}(1 - s_{i,n}), \quad (26)$$
$$x_{i,n+1} = 4x_{i,n}(1 - x_{i,n}), \quad (27)$$

Algorithm 2 Modified Quantum Particle Swarm Algorithm

Input: S_i, X_i, $i \in I_e \cup I_u$, $i \notin I_m$, D_e, D_u, $k_{\max}$, N_z, β_1, β_2, β_m, β_0

Output: S_i, X_i, $i \in I_m$

for $k = 1$ to $k_{\max}$ **do**
  for $n = 1$ to N_z **do**
    if $k = 1$ **then**
      if $n = 1$ **then**
        Randomly generate position parameters $S_{i,n,k}$ and $X_{i,n,k}$. Calculate the fitness $A_{n,k}$ based on equations (12)-(16)
        Let $P_{s,i,n} = S_{i,n,k}$, $P_{x,i,n} = X_{i,n,k}$, $P_{sg,i} = S_{i,n,k}$, and $P_{xg,i} = X_{i,n,k}$
        Set the fitness of the global optimal particle $A_g = A_{n,k}$, the current optimal fitness of particles $A_{sx,n} = A_{n,k}$
      else
        Iterate $S_{i,n,k}$ and $X_{i,n,k}$ based on equations (24)-(27) until the constraints (17a)-(17e) are met
        Calculate the fitness $A_{n,k}$ based on equations (12)-(16)
        Let $P_{s,i,n} = S_{i,n,k}$, $P_{x,i,n} = X_{i,n,k}$, and $A_{sx,n} = A_{n,k}$
        if $A_{n,k} > A_g$ **then**
          Let $P_{sg,i} = S_{i,n,k}$, $P_{xg,i} = X_{i,n,k}$
        end if
      end if
    else
      Calculate $S_{i,n,k}$ and $X_{i,n,k}$ based on equations (28)-(38), and calculate the fitness $A_{n,k}$ based on equations (12)-(16)
      while the particles satisfy the constraints (17a)-(17e) **do**
        if $A_{n,k} > A_{sx,n}$ **then**
          Let $P_{s,i,n} = S_{i,n,k}$, $P_{x,i,n} = X_{i,n,k}$
        end if
        if $A_{n,k} > A_g$ **then**
          Let $P_{sg,i} = S_{i,n,k}$, $P_{xg,i} = X_{i,n,k}$
        end if
      end while
    end if
  end for
end for
Let $S_i = P_{sg,i}$ and $X_i = P_{xg,i}$

where $i \in I_m$, $s_{i,n}$ and $x_{i,n}$ are random values between 0 and 1. The fixed points 0, 0.25, 0.5, 0.75 and 1 cannot be taken as initial values. $S_{i,n,1}$ and $X_{i,n,1}$, which are two parameters of the ith component of the nth particle of the first generation, represent the target base station selection and the relay base station selection respectively.

Step 2: calculate the fitness of each particle as the benefit obtained from migration.

Step 3: update the optimal position of each particle and the current global optimal position. If the number of iterations reaches a value or the optimal result has not been updated in a period, stop calculating. Otherwise, enter the next step.

Step 4: calculate the positions of next generation particle swarm and turn to Step 2.

The specific calculating process in step 4 is as follows:

$$S_{i,n,k+1} = \left\lceil p_{s,i,n,k} \pm \frac{L_{s,i,n,k}}{2} \ln(1/u(k)) \right\rceil \bmod N_s, \quad (28)$$

$$X_{i,n,k+1} = \left\lceil p_{x,i,n,k} \pm \frac{L_{x,i,n,k}}{2} \ln(1/u(k)) \right\rceil \bmod N_s, \quad (29)$$

$$p_{s,i,n,k} = \frac{\varphi_{s,i,n} P_{s,i,n} + \varphi_{s,i,n2} P_{sg,i}}{\varphi_{s,i,n} + \varphi_{s,i,n2}}, \quad (30)$$

$$p_{x,i,n,k} = \frac{\varphi_{x,i,n} P_{x,i,n} + \varphi_{x,i,n2} P_{xg,i}}{\varphi_{x,i,n} + \varphi_{x,i,n2}}, \quad (31)$$

where $p_{s,i,n,k}$ and $p_{x,i,n,k}$ are decision variables of ith component of the attractor of the nth particle at the kth iteration. The attractor exists to ensure the convergence of the algorithm, and each particle converges to an attractor. $P_{s,i,n}$ and $P_{x,i,n}$ are decision variables of the ith component of the optimal position of the nth particle. $P_{sg,i}$ and $P_{xg,i}$ are decision variables of the ith component of the current global optimal position of the particle swarm. $\varphi_{s,i,n}$, $\varphi_{s,i,n2}$, $\varphi_{x,i,n}$, $\varphi_{x,i,n2}$ and $u(k)$ are random factors between (0, 1). Denote L as the probability of the particle appearing at the relative point position. $L_{s,i,n,k}$ and $L_{x,i,n,k}$ are decision variables of the ith component of L of the nth particle at the kth iteration.

$$L_{s,i,n,k+1} = 2\beta(k) \left| mbest_{s,i} - S_{i,n,k} \right|, \quad (32)$$

$$L_{xi,n,k+1} = 2\beta(k) \left| mbest_{x,i} - X_{i,n,k} \right|, \quad (33)$$

where $\beta(k)$ is a very important variable, and it has a certain impact on the search ability, convergence speed and accuracy for the quantum particle swarm optimization algorithm [16]. The larger the value of β, the better search ability for the global region. However, the smaller the value of β, the better search ability for local regions. Besides, the convergence speed and the precision cannot be taken into account at the same time. Compared with the linear reduction of the number of iterations in the traditional quantum particle swarm algorithm [17], MQPS in this paper sets the number of iterations decreases according to the cosine curve. So the $\beta(k)$ is defined as

$$\beta(k) = \frac{\beta_m - \beta_0}{2} \cos\left(\frac{k\pi}{k_{\max}}\right) + \frac{\beta_m + \beta_0}{2}, \quad (34)$$

where β_m and β_0 are the upper and lower bound of β respectively. k is the current number of iterations and $k_{\max}$ is the maximum number of iterations.

In the initial stage of MQPS, due to the large difference between the global extremum of the particle and the historical individual extremum of the particle, a faster global search is used to quickly approach the global extremum. Moreover, in the later stage of the algorithm iteration, after the particle historical extremum closing to the global extremum, the smaller speed is used to enhance the local search ability and improve the accuracy of the algorithm.

It is noted that $mbest_{s,i}$ and $mbest_{x,i}$ in (32)-(33) are two decision variables of the ith component of the individual weighted average optimal position of the current particle. We change the average value of the optimal position of each particle into the random weighted average to more easily break away from local extreme point constraints as

$$mbest_{s,i} = \sum_{n=1}^{N_z} \kappa_n P_{s,i,n}, \quad (35)$$

$$mbest_{x,i} = \sum_{n=1}^{N_z} \kappa_n P_{x,i,n}. \quad (36)$$

We normalize a random number between (0, 1) to get κ_n, which indicates the contribution rate of the current optimal position of each particle to $mbest$.

The position of next generation particle equation is

$$S_{i,n,k+1} = \begin{cases} \left\lceil p_{s,i,n,k} + \beta(k) \left| mbest_{s,i} - S_{i,n,k} \right| \ln(1/u(k)) \right\rceil \\ \bmod N_s, & \text{if } u(k) \geq 0.5 \\ \left\lceil p_{s,i,n,k} - \beta(k) \left| mbest_{s,i} - S_{i,n,k} \right| \ln(1/u(k)) \right\rceil \\ \bmod N_s, & \text{if } u(k) < 0.5 \end{cases} \quad (37)$$

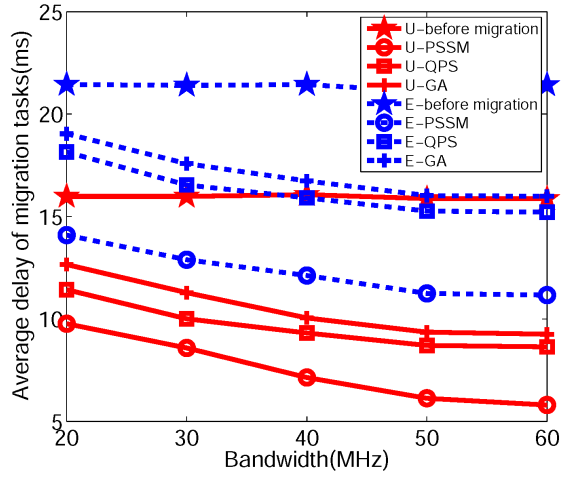
$$X_{i,n,k+1} = \begin{cases} \left\lceil p_{x,i,n,k} + \beta(k) \left| mbest_{x,i} - X_{i,n,k} \right| \ln(1/u(k)) \right\rceil \\ \bmod N_s, & \text{if } u(k) \geq 0.5 \\ \left\lceil p_{x,i,n,k} - \beta(k) \left| mbest_{x,i} - X_{i,n,k} \right| \ln(1/u(k)) \right\rceil \\ \bmod N_s, & \text{if } u(k) < 0.5 \end{cases} \quad (38)$$

IV. PERFORMANCE EVALUATION

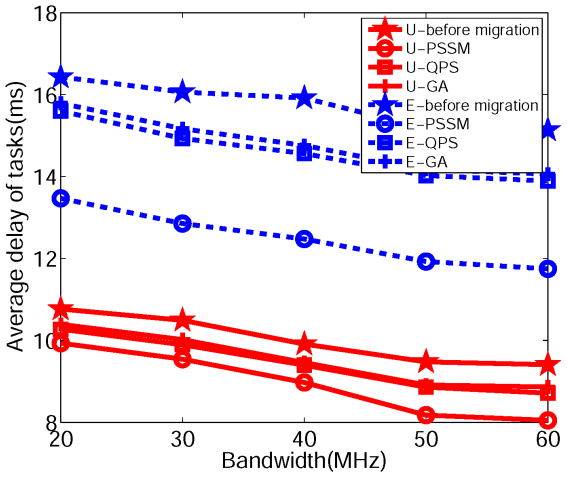
In this section, we evaluate the performance of the proposed scheme PSSM. We compare the system performance before and after service migration for eMBB tasks and uRLLC tasks under different network conditions. Moreover, we compare the migration performance of PSSM with the traditional Quantum Particle Swarm algorithm (QPS) and Greedy Algorithm(GA), in which the allocated computation resource is fixed. In QPS, $\beta(k)$ changes linearly and $mbest$ is set as the average value of the optimal position of each particle. In GA, the migration decision has two steps: target server selection and relay base station selection. First, the server with the lowest load is selected as the target server. If the base station where the target server is located can directly communicate

TABLE 1. Simulation parameters.

Simulation parameter	Simulation value
The ratio of different tasks	eMBB: 0.5, uRLLC: 0.5
N	1000
$V_{u\,\min}$ (GHz)	1.5
$V_{e\,\min}$ (GHz)	0.75
V_{server} (GHz)	12
$P_{\max}$ (W)	150
P_{idle} (W)	75
S_{task} (KB)	(100,200)
$P_{rx\,\min}$ (dbm)	-100
$P_{out\,\max}$	10^{-5}
P_{tx} (W)	1
P_{bs} (W)	20
D_e (ms)	20
D_u (ms)	15



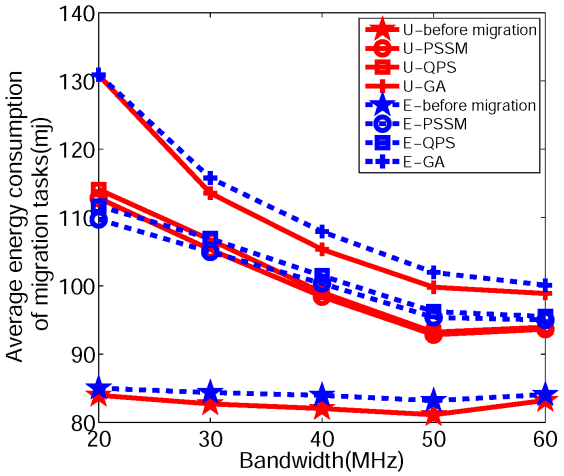
(a)



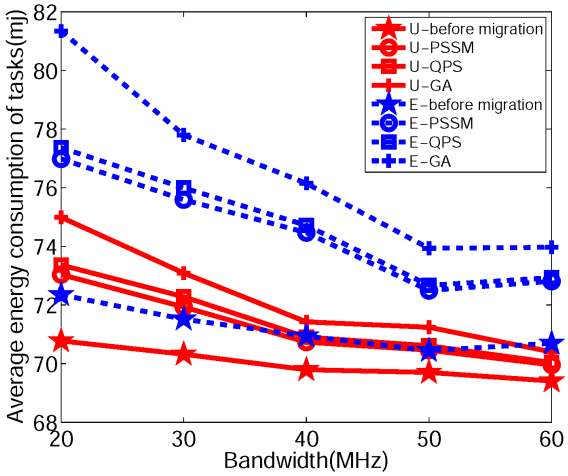
(b)

FIGURE 2. Average delay before and after migration under different bandwidths.

with the user, the relay is not needed. Otherwise, the closest base station that is directly connectable to the user is selected as the relay.



(a)



(b)

FIGURE 3. Average energy consumption before and after migration under different bandwidths.

The simulation parameters are given in Table 1. As mentioned above, QPS and GA will allocate fixed computation resource to servers. We set this value is 2 and 1 for uRLLC task and eMBB task respectively.

Fig. 2 shows the average delay of different tasks under different bandwidths. In this figure, "U-before migration" represents the delay of uLLRC tasks before migration; "U-PSSM" represents the delay of uRLLC tasks migrated by using PSSM; "U-QPS" represents the delay of uRLLC tasks migrated by using QPS; "U-GA" represents the delay of uRLLC tasks migrated by using GA. For eMBB tasks, "E-before migration", "E-PSSM", "E-QPS" and "E-GA" represent the similar meanings. Fig. 2(a) shows the average delay of tasks that need migration. Fig. 2(b) shows the average delay of all tasks. In Figs. 2(a) and (b), for two kinds of tasks, all migration algorithms can effectively reduce the delay and PSSM can get the best result. From Fig. 2(a), we can see that the larger the bandwidth, the smaller the average delay after migration. In particular,

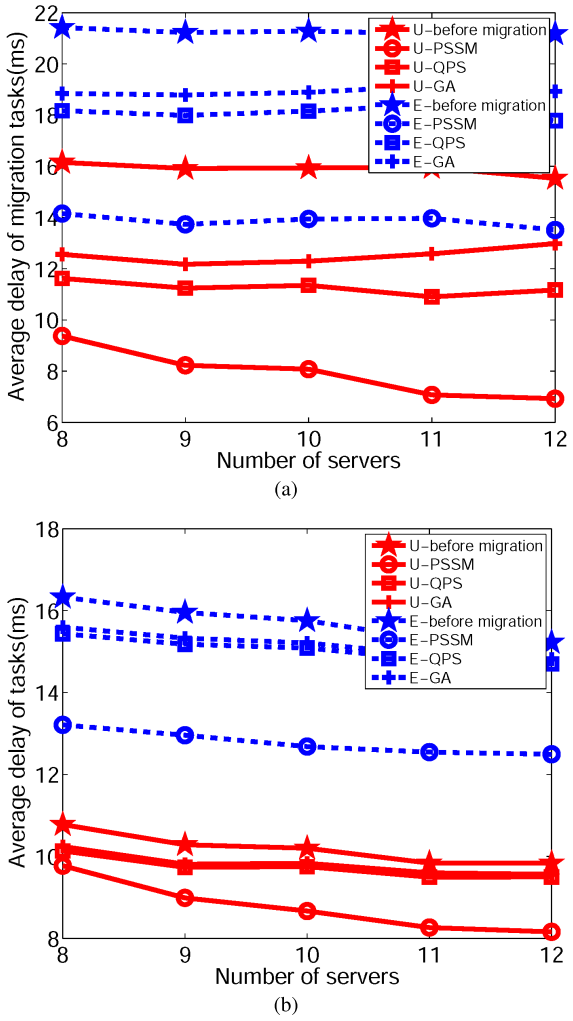


FIGURE 4. Average delay before and after migration under different number of servers.

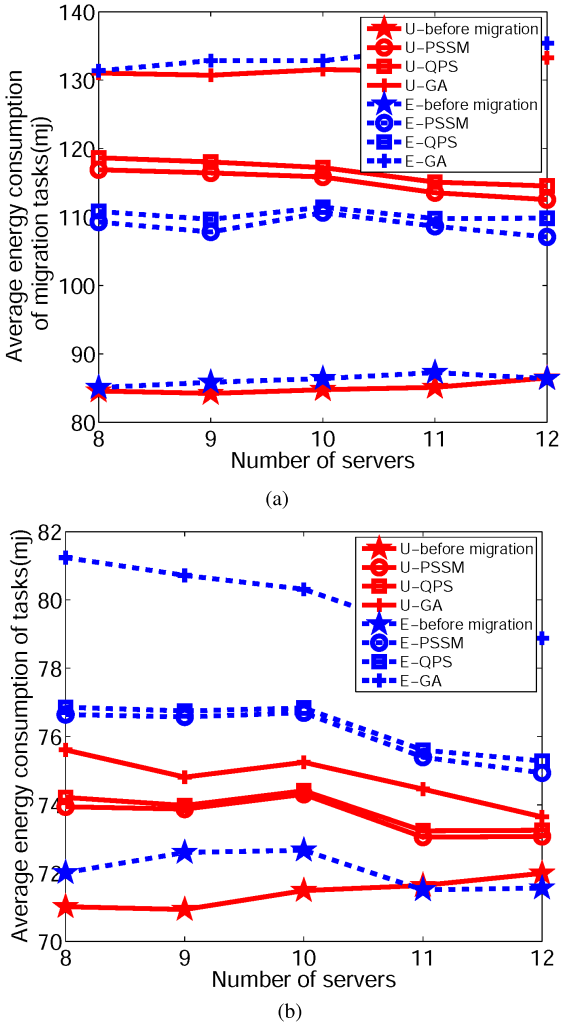


FIGURE 5. Average energy consumption before and after migration under different number of servers.

the transmission delay and the relay delay will decrease as the bandwidth increases. The processing delay and queuing delay of the tasks that need to be migrated will become higher as the bandwidth increases. Nevertheless, the migration can reduce processing delay and queuing delay, which makes the average delay decrease more drastically under high bandwidth.

Fig. 3(a) shows the average energy consumption of tasks that need migration. Fig. 3(b) shows the average energy consumption of all tasks. Although the migration will bring extra energy consumption, the proposed PSSM has better performance than QPS and GA. Moreover, since the energy consumption for queuing is not considered, the processing energy consumption is only related to the size of the task, and then the transmission delay, relay delay and migration delay are inversely proportional to the bandwidth.

Fig. 4 shows the average delay under different number of servers. In Fig. 4(a), all migration algorithms can effectively reduce delay and PSSM can get the best result. Besides, with the increase of the number of servers, the advantage of PSSM becomes more obvious. Most of the time, the sub-algorithm DCRA in PSSM can allocate more computation resource to tasks, which can greatly reduce processing delay and queuing delay. When the number of servers increases, there are more targets for migration, so the transmission delay, relay delay and migration delay also decrease.

Fig. 5 shows the average energy consumption under different number of servers. In Figs. 5(a) and (b), we can see that the energy consumption of PSSM is lower than QPS and GA. This indicates that the performance of PSSM is not affected by the number of servers.

V. CONCLUSION

In the mobile edge computing circumstance, as the mobility of users and the limited resources of edge nodes, some edge nodes cannot provide high quality services. In view of this, a Particle Swarm based Service Migration scheme (PSSM) has been proposed to migrate services from the initial nodes to other edge nodes to meet needs of users. PSSM consists of Queuing Delay Prediction algorithm (QDP),

Delay-aware Computation Resource Allocation algorithm (DCRA), and Modified Quantum Particle Swarm algorithm (MQPS). Specifically, QDP is designed to predict the queuing delay of tasks and DCRA is designed to allocate computation resource dynamically to improve the efficiency of servers. By taking advantages of quantum particle swarm optimization, we design MQPS to address the service migration. The simulation results show that our scheme PSSM can effectively reduce delay and energy consumption, but also improve the processing capability of servers.

REFERENCES

[1] 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, doi: 10.1109/ACCESS.2018.2828102.

[2] N. Abbas, Y. Zhang, A. Taherkordi, and T. Skeie, "Mobile edge computing: A survey," *IEEE Internet Things J.*, vol. 5, no. 1, pp. 450–465, Feb. 2018, doi: 10.1109/JIOT.2017.2750180.

[3] F. Zhang, G. Liu, X. Fu, and R. Yahyapour, "A survey on virtual machine migration: Challenges, techniques, and open issues," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 2, pp. 1206–1243, 2nd Quart., 2018, doi: 10.1109/COMST.2018.2794881.

[4] S. Wang, X. Zhang, Y. Zhang, L. Wang, J. Yang, and W. Wang, "A survey on mobile edge networks: Convergence of computing, caching and communications," *IEEE Access*, vol. 5, pp. 6757–6779, 2017, doi: 10.1109/ACCESS.2017.2685434.

[5] N. Lu, N. Cheng, N. Zhang, X. Shen, and J. W. Mark, "Connected vehicles: Solutions and challenges," *IEEE Internet Things J.*, vol. 1, no. 4, pp. 289–299, Aug. 2014, doi: 10.1109/JIOT.2014.2327587.

[6] M. Nelson, B.-H. Lim, and G. Hutchins, "Fast transparent migration for virtual machines," in *Proc. USENIX Annu. Tech. Conf.*, Anaheim, CA, USA: USENIX Association, Apr. 2005, p. 25.

[7] L. F. Bittencourt, M. M. Lopes, I. Petri, and O. F. Rana, "Towards virtual machine migration in fog computing," in *Proc. 10th Int. Conf. P2P, Parallel, Grid, Cloud Internet Comput. (3PGCIC)*, Kraków, Poland, Nov. 2015, pp. 1–8, doi: 10.1109/3PGCIC.2015.85.

[8] A. Machen, S. Wang, K. K. Leung, B. J. Ko, and T. Salonidis, "Live service migration in mobile edge clouds," *IEEE Wireless Commun.*, vol. 25, no. 1, pp. 140–147, Feb. 2018, doi: 10.1109/MWC.2017.1700011.

[9] D. Zhao, T. Yang, Y. Jin, and Y. Xu, "A service migration strategy based on multiple attribute decision in mobile edge computing," in *Proc. IEEE 17th Int. Conf. Commun. Technol. (ICCT)*, Oct. 2017, pp. 986–990, doi: 10.1109/ICCT.2017.8359782.

[10] A. Ksentini, T. Taleb, and M. Chen, "A Markov decision process-based service migration procedure for follow me cloud," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Sydney, NSW, Australia, Jun. 2014, pp. 1350–1354, doi: 10.1109/ICC.2014.6883509.

[11] W. Zhang, Y. Hu, Y. Zhang, and D. Raychaudhuri, "SEGUE: Quality of service aware edge cloud service migration," in *Proc. IEEE Int. Conf. Cloud Comput. Technol. Sci. (CloudCom)*, Luxembourg, U.K., Dec. 2016, pp. 344–351, doi: 10.1109/CloudCom.2016.0061.

[12] S. Wang, R. Urgaonkar, M. Zafer, T. He, K. Chan, and K. K. Leung, "Dynamic service migration in mobile edge-clouds," in *Proc. IFIP Netw. Conf. (IFIP Netw.)*, Toulouse, France, May 2015, pp. 1–9, doi: 10.1109/IFIPNetworking.2015.7145316.

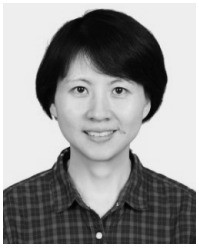
[13] S. Wang, R. Urgaonkar, T. He, M. Zafer, K. Chan, and K. K. Leung, "Mobility-induced service migration in mobile micro-clouds," in *Proc. IEEE Mil. Commun. Conf.*, Baltimore, MD, USA, Oct. 2014, pp. 835–840, doi: 10.1109/MILCOM.2014.145.

[14] T. Ojima and T. Fujii, "Resource management for mobile edge computing using user mobility prediction," in *Proc. Int. Conf. Inf. Netw. (ICOIN)*, Chiang Mai, Thailand, Jan. 2018, pp. 718–720, doi: 10.1109/ICOIN.2018.8343212.

[15] R. M. Golubovic, D. I. Olcan, and B. M. Kolundzija, "Particle swarm optimization algorithm and its modifications applied to EM problems," in *Proc. 8th Int. Conf. Telecommun. Modern Satell., Cable Broadcast. Services*, Nis, Serbia, Sep. 2007, pp. 427–430, doi: 10.1109/TELSKS.2007.4376029.

[16] J. Sun, B. Feng, and W. Xu, "Particle swarm optimization with particles having quantum behavior," in *Proc. Congr. Evol. Comput.*, Portland, OR, USA, vol. 1, Jun. 2004, pp. 325–331, doi: 10.1109/CEC.2004.1330875.

[17] J. Sun, W. Xu, and B. Feng, "Adaptive parameter control for quantum-behaved particle swarm optimization on individual level," in *Proc. IEEE Int. Conf. Syst., Man Cybern.*, Waikoloa, HI, USA, vol. 4, Oct. 2005, pp. 3049–3054, doi: 10.1109/ICSMC.2005.1571614.



LIANG LIANG received the B.Eng. and M.Eng. degrees from the Southwest University of Science and Technology (SWUST), China, in 2003 and 2006, respectively, and the Ph.D. degree in communication and information system from the University of Electronic Science and Technology of China (UESTC), in 2012. From August 2011 to January 2012, she was an International Visitor with the Institute for Infocomm Research (I2R), Singapore. She is currently an Associate Professor with the School of Microelectronics and Communication Engineering, Chongqing University, Chongqing, China. Her research interests include wireless communication and optimization, wireless network virtualization, mobile edge computing, and the IoT.

JINTAO XIAO received the B.Eng. degree in electronic information engineering from Chongqing University, Chongqing, China, in 2017, where he is currently pursuing the M.Eng. degree. His current research interests include wireless communication and edge computing.

ZHI REN received the B.S. degree in applied electronics from Southwest Jiaotong University, Chengdu, China, in 1993, and the M.S. degree in measuring and testing technology and the Ph.D. degree in communication and information systems from the University of Electronic Science and Technology of China, in 2002 and 2005, respectively. From 2006 to 2008, he was a Postdoctoral Research Associate with the Department of Electrical and Computer Engineering, Stevens Institute of Technology, NJ, USA. He is currently a Professor with the School of Communication and Information Engineering, Chongqing University of Posts and Telecommunications, China. His research interests include wireless mobile networks and wireless communications.

ZHENGCHUAN CHEN (Member, IEEE) received the B.S. degree from Nankai University, China, in 2010, and the Ph.D. degree from Tsinghua University, China, in 2015.

He visited The Chinese University of Hong Kong, in 2012, and visited the University of Florida, USA, in 2013. From 2015 to 2017, he was a Postdoctoral Fellow of the Information Systems Technology and Design Pillar, Singapore University of Technology and Design (SUTD). He is currently an Assistant Professor with the School of Microelectronics and Communication Engineering, Chongqing University, China. His main research interests include ultrareliable and low latency communications, age of information, and network information theory. He has served several IEEE conferences, e.g., the IEEE Globecom, as a Technical Program Committee Member. He co-received the Best Paper Award from the International Workshop on High Mobility Wireless Communications, in 2013. He was selected as an Exemplary Reviewer of the IEEE TRANSACTIONS ON COMMUNICATIONS, in 2015.

YUNJIAN JIA received the B.S. degree from Nankai University, China, in 1999, and the M.E. and Ph.D. degrees in engineering from Osaka University, Japan, in 2003 and 2006, respectively.

From 2006 to 2012, he was a Researcher with the Central Research Laboratory, Hitachi Ltd., where he engaged in research and development on wireless networks, and contributed to LTE and LTE-Advanced standardization in 3GPP. He is currently a Professor with the School of Microelectronics and Communication Engineering, Chongqing, China. He is the author of more than 90 published articles and the inventor of more than 30 granted patents. His research interests include future radio access technologies, mobile networks, and the IoT. He received several prizes from industry and academia, including the IEEE Vehicular Technology Society Young Researcher Encouragement Award, the IEICE Paper Award, the APCC2017 Best Paper Award, the China Industry-University-Research Institute Collaboration Innovation Award, the Yokosuka Research Park R&D Committee YRP Award, and the Top 50 Young Inventors of Hitachi. Moreover, he was a recipient of the Research Fellowship Award in International Communication Foundation and Telecommunications Advancement Foundation, Japan.