Hindawi
Mathematical Problems in Engineering
Volume 2020, Article ID 4124791, 17 pages
https://doi.org/10.1155/2020/4124791

Research Article

A Multilevel Optimization Framework for Computation Offloading in Mobile Edge Computing

Nanliang Shan, Yu Li, and Xiaolong Cui

College of Information Engineering, Engineering University of PAP, Xi'an 710086, China

Correspondence should be addressed to Xiaolong Cui; shannanliang@126.com

Received 20 April 2020; Revised 30 May 2020; Accepted 5 June 2020; Published 27 June 2020

Guest Editor: Qiang He

Mobile edge computing is a new computing paradigm that can extend cloud computing capabilities to the edge network, supporting computation-intensive applications such as face recognition, natural language processing, and augmented reality. Notably, computation offloading is a key technology of mobile edge computing to improve mobile devices' performance and users' experience by offloading local tasks to edge servers. In this paper, the problem of computation offloading under multiuser, multiserver, and multichannel scenarios is researched, and a computation offloading framework is proposed that considering the quality of service (QoS) of users, server resources, and channel interference. This framework consists of three levels. (1) In the offloading decision stage, the offloading decision is made based on the beneficial degree of computation offloading, which is measured by the total cost of the local computing of mobile devices in comparison with the edge-side server. (2) In the edge server selection stage, the candidate is comprehensively evaluated and selected by a multiobjective decision based on the Analytic Hierarchy Process based on Covariance (Cov-AHP) for computation offloading. (3) In the channel selection stage, a multiuser and multichannel distributed computation offloading strategy based on the potential game is proposed by considering the influence of channel interference on the user's overall overhead. The corresponding multiuser and multichannel task scheduling algorithm is designed to maximize the overall benefit by finding the Nash equilibrium point of the potential game. Amounts of experimental results show that the proposed framework can greatly increase the number of beneficial computation offloading users and effectively reduce the energy consumption and time delay.

1. Introduction

With the development of artificial intelligence and Internet of Things (IoT) technology, a large number of computation-intensive and time-sensitive mobile applications have appeared on mobile terminals, such as face recognition, natural language processing, and augmented reality. However, the limited computing capability and energy storage of mobile terminals cannot provide intensive computing and complete high-energy tasks. A large number of applications training process must be deployed on the cloud service platform, so that a large amount of training data generated on the mobile terminal need to be transmitted to the cloud through the core network, resulting in a sharp increase in the already congested core network load. Application transmission delay and transmission energy consumption will also increase greatly, which will cause service request failure and QoS of the user to decline.

Currently, mobile edge computing (MEC) [1–3] has become an important solution to the above contradictions. However, in the MEC scheme based on the public cloud [3–5], accessing the mobile cloud service through the wireless channel results in a large channel blocking rate and delay. Meanwhile, the MEC scheme based on the cloudlet [6] can only connect to cloud services via WiFi, which has a large space limitation.

In response to the above limitations, an edge server-based MEC solution has been proposed and widely used. As shown in Figure 1, the edge server is deployed on the wireless LAN side in the edge server-based MEC solution, which shortens the distance between servers and the users. This solution can use computing offload to expand the service

capabilities of mobile devices, provide localized computing and storage resources for mobile devices nearby, reduce data transmission costs, and meet the needs of the fast and interactive response. Among them, mobile devices and users are collectively referred to as mobile edge nodes, and the combination of wireless base stations and edge servers is referred to as edge server nodes. The edge server nodes can further offload the preprocessing results to the remote cloud service center through the mobile core network. Therefore, the key technology of the MEC solution based on the edge server is computing offloading. How to formulate a reasonable and efficient computing offloading framework will be an issue that we urgently need to solve. When many users offload computing tasks to the same edge server through the same channel at the same time, it will cause congestion and greater delay. The evaluation indicators for computing offload include energy consumption and service delay.

In order to minimize the energy consumption and service delay of mobile users, the main contributions of this paper are listed as follows:

(1) The concept of beneficial computation offloading is proposed. The overall cost of the mobile edge computing system is defined as the weighted sum of energy consumption, the corresponding transmission delay of computation offloading, and the task processing of all edge nodes. When the total cost of computation offloading is less than the total cost of the local computing, beneficial computation offloading is obtained. Beneficial computation offloading is a prerequisite for users to make computation offloading decisions.

(2) A Cov-AHP strategy for multiobjective servers offloading decision is proposed. Considering the transmission time, energy consumption, and server residual resources, the server to be selected for undertaking offloading is comprehensively evaluated by the covariance judgment matrix. The experimental results show that the Cov-AHP algorithm is objective and can effectively realize load balancing.

(3) A multiuser multichannel distributed computation offloading algorithm based on the potential game is proposed. The computation offloading optimization of multiuser in a multichannel wireless interference scenario is an NP-Hard problem. According to the number of beneficial computation offloading users and the total system cost, the efficiency of the Nash equilibrium is quantified. Users can formulate an interaction mechanism based on the group strategy to achieve the maximum benefit and the highest resource utilization. The algorithm can optimize its computation offloading performance as the size of the user increases.

The remainder of this paper is organized as follows. Section 2 reviews the related studies. Section 3 introduces the system model and describes the problem statement and solution. In Sections 4, 5, and 6, a three-level strategy for the computation offloading framework is illustrated in detail. The experiment and result analysis are discussed in Section 7. The final section summarizes the paper and discusses future work.

2. Related Studies

Researches on computation offloading can be divided into three types. In the mode of single-user single-available edge server, a dynamic computation migration algorithm (LODCO) based on Lyapunov optimization [7] is proposed to optimize the execution delay of the application. In [8], the optimization of mobile device energy is defined as a constrained Markov decision process. Then, a computational migration decision algorithm is proposed to equalize the execution delay and the energy consumption of the mobile terminal. The literature [9] defined the computational migration decision problem as a nonconvex quadratic constrained programming and proposed a semicustom random heuristic algorithm to significantly reduce the overall cost of the system. In order to balance the energy consumption and delay during the migration process, the literature [10] adopted cost function to measure the migration request aggregation and proposed an online algorithm considering both energy consumption and QoS. A deep learning model based on network packet information to evaluate the quality of experience (QoE) of users and servers was proposed in [11].

In the mode of multiuser single-available edge server, an offloading algorithm based on distributed deep learning is proposed [5], which can provide approximate optimal offloading decisions for multiple mobile device users and single edge servers in MEC. A classification-based energy-saving computation migration algorithm (EECO) is proposed in [3, 12], which can save about 15% energy consumption in comparison. A three-step algorithm is proposed to design the semideterministic relaxation (SDR), alternating optimization (AO), and sequential adjustment (SA) strategies [13] to achieve a joint optimization scheme of computing resources and communication resources. A novel approach to formulate cost-efficient fault tolerance strategies for multitenant service-based systems is proposed in [14].

In the mode of multiuser multiavailable edge servers, the linear relaxation method and the semidetermined relaxation (SDR) method are used to minimize the total task execution delay and energy consumption in the MEC with the multiedge servers [15]. The best computation distribution between multiple edge servers is given in [16]. In order to maximize the long-term utility, a model-free reinforcement learning offloading mechanism (Q-learning) is proposed in [17]. An adaptive computation offloading method with both macrobase stations (MABSs) in 5G and roadside units (RSUs) in Internet of Connected Vehicles (IoCV) to optimize the task execution delay and energy consumption of the edge system is proposed in [18]. A two-phase computation offloading optimization method to maximize the resource utilization of ECUs, minimize the execution time, and balance the privacy preservation and execution performance is proposed in [19]. A blockchain-enabled computation offloading method to achieve tradeoffs among minimizing

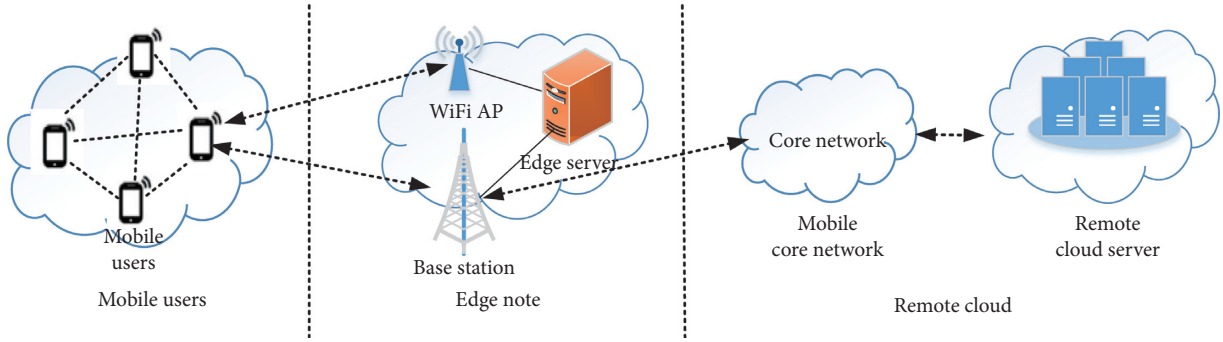


Figure 1: Mobile edge computing (MEC) architecture

ECDs' task offloading time, optimizing energy consumption, and maintaining load balance for IoT is devised in [20]. An optimal approach to solve the dynamic QoS edge user allocation (EUS) problem and a heuristic approach for quickly finding suboptimal solutions to large-scale instances of the dynamic QoS EUS problem are proposed in [21].

Additionally, multichannel radio interference has aroused much attention. The literature [3] utilizes game theory to realize efficient distributed channel allocation and computation offloading in multichannel wireless interference environments. A two-layer optimization method based on orthogonal frequency division multiplexing (OFDM) is proposed in [22, 23] to solve the problem of subcarrier allocation and task offloading of multiuser access multi-available edge servers. A theorem method to describe edge user allocation (EUA) problem as potential games and a novel decentralized algorithm that can solve EUA problem effectively are proposed in [24]. An optimal method to deal with EUA problem is proposed in [25]. The EUA problem is modelled as a bin packing problem, and the Lexicographic Goal Programming technique is adopted. An metric that can measure the community-diversified influence is proposed in [26]. Three novel QoS-aware service selection approaches for composing multitenant SBSs that achieve three different multitenancy maturity levels is presented in [27]. A novel strategy CFT4MTS (Criticality-Based Fault Tolerance for Multi-Tenant SBSs) that formulates cost-effective fault tolerance for multitenant SBSs by providing redundancy for the critical component services is proposed in [14]. A novel Web API recommendation method called keywords-based and compatibility-aware API recommendation based on weighted API correlation graph is proposed in [28]. A multidimensional quality ensemble-driven recommendation method based on the Locality-Sensitive Hashing technique and Order Preference by Similarity to Ideal Solution technique is proposed in [29]. The original stochastic problem is transformed to the determinist optimization problem by adopting stochastic optimization techniques, and an energy efficient dynamic offloading algorithm is proposed in [30]. A blockchain-based computation offloading method for edge computing in 5G networks is proposed in [31].

In summary, single-user scenarios are an idealized abstraction of computation offloading. While the multiuser, multiserver, and multichannel scenarios are so complicated that involve multiusers' offloading decisions, computing resource requirements, edge server selection, and multiple channel interferences. This paper focuses on the problem of computation offloading under multiuser, multiserver, and multichannel scenarios and proposes a computation offloading framework considering the QoS of users, server resources, and channel interference. This framework consists of three stages: (1) offloading decision stage; (2) server selection stage; (3) channel selection stage. In the corresponding stage, the corresponding solution model is proposed to optimize the user's decision.

3. System Model and Problem Statement

In the Mobile Edge Cloud Service model, we consider setting up N mobile users and K edges server nodes, where each user has a computation-intensive task [32] and each edge server node is composed of a base station and an edge server. Mobile users can offload computing tasks to edge servers through the base station. We also consider setting M wireless channels between mobile users and each edge server node. Table 1 lists the parameters used in this paper. Computing tasks that are offloaded to the same edge server node will be transmitted over different channels. The mobile users evaluate the computing task characteristics, energy reserve, computing capability, and network communication quality to make computation offloading decisions, which are divided into local computing and edge server-side computing. In the edge server-side computing mode, mobile users search for a suitable edge server, for which computing tasks offloading decisions is beneficial [33]. We call this beneficial computation offloading, which can reduce the energy consumption, shorten the delay, and guarantee the QoS of users. Then, we search a channel with high bandwidth and low interference to the selected edge server. The computation task is offloaded to the edge server, and the result is returned to the edge user through the channel. Among them, communication and computation are the two most important components of the Mobile Edge Cloud Service model. And, the model is described as follows.

In order to effectively represent different scenarios in the edge server-based MEC scheme, this paper proposes a Mobile Edge Cloud Service model, including the communication model and computation model.

Table 1: Parameter description.

Parameter	Description
M	Number of channels
k	Number of the edge servers
k'	Number of wireless base-stations
B_n	Size of computation task
D_n	Number of CPU cycles required to the computation task
f_n^l	Computing speed of user CPU
f_n^v	Computing speed of single VM CPU
f_n^c	Computing speed of edge server CPU
γ_n^l	Consumed energy per CPU cycle
λ_n^t	Weight of computational time
λ_n^e	Weight of energy consumption
β	Distance between users and the base station
α	Channel loss factor
g_n	Channel gain
q_n	Transmission power
σ	Background noise power
K	Channel bandwidth

3.1. Mobile Edge Cloud Service Communication Model. Each edge-end user can process tasks locally or offload computing tasks to the edge server over a wireless channel. Assume that the number of edge-end users that may perform computational offloading is N and there are M wireless channels between the edge-end user and the edge server node. User n's computational offload strategy a_n is defined as

$$a_n = \begin{cases} 0, & \text{User } n \text{ processes tasks locally,} \\ 1, & \text{User } n \text{ unloads the task to the edge server.} \end{cases} \tag{1}$$

Then, $a_N = (a_1, a_2, \ldots, a_n)$ is the set of computation offloading decisions for all users. According to the Shannon spectrum formula, the edge user offloads the computing task to the edge server, and the uplink transmission rate is given as

$$C_n(a_N) = Klog_2\left(1 + \frac{S_n}{N_i}\right), \tag{2}$$

where $S_n = q_n g_n$, q_n and g_n are the transmission energy and channel gain of the user n communicating with the base station, K is the channel bandwidth of the wireless transmission process, and $N_i = \sigma + \sum_{i\in\{N\}:\ a_i=a_n} q_i g_i$, in which σ is the background white noise interference and $\sum_{i\in\{N\}:\ a_i=a_n} q_i g_i$ is the communication interference of other channels. The channel capacity calculated by Shannon theorem is the maximum available data transmission rate. Usually, the actual data transmission rate is less than the channel capacity. Using Shannon theorem to evaluate the achievable data transmission rate, the minimum delay of data transmission can be obtained. This minimum delay value can be used as a threshold for offloading decisions.

3.2. Mobile Edge Cloud Service Computation Model. The computation model includes local computing and edge server-side computing. The compute-intensive tasks for edge users are defined as $I_n \triangleq (B_n, D_n)$, where B_n is the data amount that the user needs to complete the task [34] and D_n is the number of CPU clock cycles required to complete the task [35].

3.2.1. Local Computing. When the edge user's decision is local computing, the edge user n executes the computation task I_n locally. Assume f_n^l is the computing capability of the edge user (the clock frequency unit of the edge user CPU runs is HZ). γ_n^l is the energy consumption of each CPU cycle, which can be obtained by the measurement method in [36]. So, we can get the execution time and energy consumption of the local computing task I_n as

$$\begin{aligned} t_n^l &= \frac{D_n}{f_n^l}, \\ e_n^l &= \gamma_n^l D_n. \end{aligned} \tag{3}$$

For the total cost of the local computing task, we have that

$$K_n^l = \lambda_n^t t_n^l + \lambda_n^e e_n^l. \tag{4}$$

Among them, $\lambda_n^t, \lambda_n^e \in (0, 1)$ indicates the weights of the computation time and energy consumption given by the edge user n in the decision-making, respectively. The user can flexibly set the two weights according to the requirements of the energy consumption and sensitivity of the delay in the scenario, thereby dynamically adjusting the overall cost of the system.

3.2.2. Edge Server Side Computing. When the edge-side user's decision is the edge server-side computing, the edge-side user n offloads the computation-intensive task to the edge server through the wireless channel, and the time and energy consumption overhead of the offload transmission process is defined as

$$t_{n,\text{up}}^c(a_N) = \frac{B_n}{c_n(a_N)}, \tag{5}$$

$$e_{n,\text{up}}^c(a_N) = \frac{q_n B_n}{c_n(a_N)} + T_n. \tag{6}$$

Among them, T_n is the tailing energy generated in wireless transmission. On the edge server-side, the computing capability of the edge server is the clock frequency f_n^c, then the time of computing tasks I_n performs on the edge server node can be given as

$$t_{n,\text{exe}}^c = \frac{D_n}{f_n^c}. \tag{7}$$

According to (5), (6), and (7), we can compute the total cost of edge server-side computations as

$$K_n^c(a_N) = \lambda_n^t\left(t_{n,\text{up}}^c(a_N) + t_{n,\text{exe}}^c\right) + \lambda_n^e e_{n,\text{up}}^c(a_N). \tag{8}$$

Among them, $\lambda_n^t, \lambda_n^e \in (0, 1)$, the time and energy cost of sending computation results back to the edge node from the

edge server node [22] is ignored. Because for many intensive computing applications that need to be offloaded (e.g., face recognition and virtual reality), the size of the data set that is fed back to the user is often several orders of magnitude smaller than the size of the input data set.

3.3. Problem Statement and Solution. Therefore, the computation offloading of the Mobile Edge Cloud Service model involves three problems:

(1) How to decide whether the computing task is completed locally or offloaded to the edge server?

(2) How to determine the appropriate edge server for computation offloading?

(3) How to choose the right channel to achieve the highest wireless transmission efficiency?

In order to solve the above problems, the computation offloading optimization framework proposed in this paper consists of three levels.

3.3.1. Offloading Decision Stage. In this level, we propose a beneficial computation offloading decision strategy. And, according to the strategy, the overall cost of edge user local computing and edge server-side computing is compared. Then, the computing mode is determined based on the beneficial degree of computation offloading.

3.3.2. Edge Server Selection Stage. After the edge user makes the decision to computation offloading, we propose a server selection strategy, which considers the transmission time, transmission energy consumption, and remaining CPU resources of the edge server. Then, we use the Cov-AHP multiobjective decision method to evaluate and select the offload table edge server according to the final weight.

3.3.3. Channel Selection Stage. To solve the problem of signal interference caused by multiple users simultaneously selecting the same edge server for computation offloading, we propose a multiuser multichannel distributed computation offloading strategy based on the potential game. In detail, the Nash equilibrium point is defined as the optimal solution for the combined optimization problem of channel selection and beneficial computation offloading.

4. Offloading Decision Stage

At this level, we present the beneficial computation offloading decision strategy. The weighted sum of the energy consumption of the edge user to offload and process the computing task and the corresponding transmission and processing delay is defined as the total cost of the system for the edge node to complete the task. We propose a beneficial computation offloading decision strategy, which minimizes the overall cost of the system by optimizing the task offloading decision a_n.

Definition 1. Beneficial computation offloading: if and only if the overall cost of edge server-side computing is less than the total cost of local computing, we call such computing offloading beneficial to the user, i.e.,

$$K_n^e(a_N) < K_n^l. \tag{9}$$

Here, we construct an indicator function $I_{\{A\}}$. When event A is true, then $I_{\{A\}} = 1$; otherwise, $I_{\{A\}} = 0$. The edge-user offloading decision $a_n = 1$ if and only if the computing task satisfies a beneficial offload.

In summary, the beneficial computation offloading problem boils down to maximizing the number of users performing beneficial edge computation and minimizing the overall cost of performing all computational tasks, so the objective function and constraints are defined as

$$\begin{cases} \max \sum_{n \in N} I_{\{a_n=1\}}, \\ K_n^c(a_N) < K_n^l, \\ \min\limits_{\{a_n\}} \sum\limits_{n=1}^{N} \left(K_n^l(1-a_n) + K_n^c(a_N)a_n\right), \\ a_n \in \{0,1\}, \quad n \in N. \end{cases} \tag{10}$$

5. Edge Server Selection Stage

At this level, we present the Cov-AHP- (Analytic Hierarchy Process Based on Covariance-) based server selection strategy. When the task satisfies the beneficial computation offloading, we consider the transmission energy consumption, the transmission delay, and the remaining resources of the edge server to select the server that meets the computation offloading condition. The novelty of the Cov-AHP-based method is that the feasibility of its evaluation scheme no longer depends on the experience of experts but on the relationship between the scheme itself and the target layer to judge. This approach can greatly reduce the subjectivity, and at the same time objectively evaluate the connection between various schemes and goals.

Firstly, we need to address the evaluation of server selection. In order to overcome the influence of human subjective judgment, this paper builds a judgment matrix based on covariance and the Cov-AHP-based server selection strategy [37] whose calculation results and sorting are unique. The strategy includes four steps.

5.1. Establishing a Hierarchical Structure. The Cov-AHP-based server selection strategy is established according to the progressive order of the target layer, the criteria layer, and the scheme layer. The Cov-AHP-based server selection strategy is shown in Figure 2.

The target layer is the ultimate goal that the strategy will ultimately achieve. In this paper, our ultimate goal is to choose the most suitable server for computing offload. The criteria layer is the element that depends on the evaluation of the server. We select three elements: transmission delay, transmission energy consumption, and remaining CPU

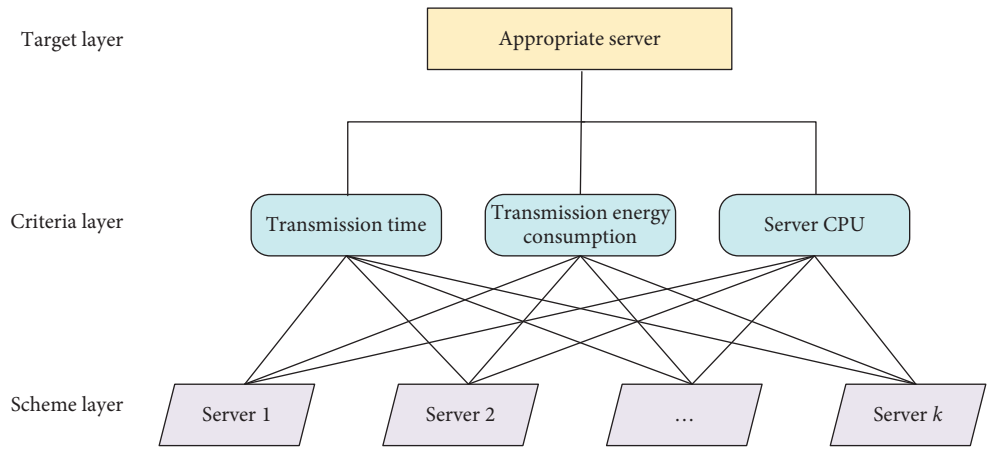


Figure 2: Cov-AHP-based server selection strategy.

resources of the server. The scheme layer is the servers that can be used for computing offload.

5.2. Constructing Judgment Matrix. For a system that has k alternatives, there is a certain objective relationship between its constituent elements, which can be expressed by covariance. The basic idea of Cov-AHP is to construct a judgment matrix reflecting the relative importance of each element. Then, based on the covariance matrix, we can obtain the weight of the relative importance of quantitative indicators between the relative layers during the analytic hierarchy process.

Suppose that the value of scheme 1 corresponding to element a is x_1 and the value of scheme 2 corresponding to element a is x_2, then the covariance of x_1 and x_2 is c_{12} and we have $c_{ij} = c_{ji}$. The covariance matrix A of element a is expressed as

$$A = \begin{pmatrix} c_{11} & \cdots & c_{1k} \\ \vdots & \ddots & \vdots \\ c_{k1} & \cdots & c_{kk} \end{pmatrix}. \tag{11}$$

Using the covariance c_{ij} of each column divide covariance c_{ii} and then taking the transformation of the product of all paired elements into one, i.e., $b_{ij} = a_{ij}/\sqrt{a_{ij} \times a_{ji}}$, finally, the judgment matrix B is constructed as

$$B = \begin{pmatrix} 1 & \cdots & b_{1k} \\ \vdots & \ddots & \vdots \\ b_{k1} & \cdots & 1 \end{pmatrix}. \tag{12}$$

Then, use the judgment matrix to calculate the weight of each element. The principle of the analytic hierarchy process shows that the eigenvector corresponding to the largest eigenvalue of the judgment matrix B is the weight vector of each element. The square root method is used to solve the feature vector as follows:

(i) Calculate the product of each row element of the judgment matrix B: $M_i = \prod_{j=1}^{k} b_{ij}$

(ii) Calculate the k-th root of each line M_i: $\omega_i = \sqrt[k]{M_i}$

(iii) Normalize and then get the weight of each element: $\omega_\alpha = \omega_i / \sum_{i=1}^{k} \omega_i$

Then, the weight vector of the B matrix can be obtained as $\omega_a = (\omega_1, \omega_2, \ldots, \omega_k)'$.

5.3. Consistency Test. Multiply the judgment matrix B by the weight vector $\omega_a = (\omega_1, \omega_2, \ldots, \omega_k)'$ of B matrix to obtain a k-order column vector BW, and then, according to the formula, $\lambda_{\max} = (1/k) \sum_{i=1}^{k} ((\mathrm{BW})_i/\omega_i)$, we can get the largest eigenvalue $\lambda_{\max}$ of the judgment matrix B. Among them, $(\mathrm{BW})_i$ represents the i-th component of the column vector BW.

The indicator CI that measures the deviation of the judgment matrix B is calculated as

$$\mathrm{CI} = \frac{\lambda_{\max} - k}{k - 1}. \tag{13}$$

The random consistency ratio CR is calculated as

$$\mathrm{CR} = \frac{\mathrm{CI}}{\mathrm{RI}}. \tag{14}$$

Among them, CR is a random consistency standard (Table 2).

When $\mathrm{CR} < 0.10$, it is generally considered that the judgment matrix B has satisfactory consistency; otherwise, the judgment value needs to be adjusted until the consistency check is passed.

5.4. Weight Integration and Server Selection. Suppose the weights of the elements of the criteria layer for the target layer are usually set according to the user needs of the edge user, the weight of each scheme relative to each element of the criteria layer is $\omega_a = (\omega_{1a}, \omega_{2a}, \ldots, \omega_{ka})'$, $\omega_b = (\omega_{1b}, \omega_{2b}, \ldots, \omega_{kb})'$, and $\omega_c = (\omega_{1c}, \omega_{2c}, \ldots, \omega_{kc})'$. The weight of each scheme relative to the target layer is

Table 2: Random consistency standard.

k	1	2	3	4	5	6	7	8	9	10
RI	0.00	0.00	0.58	0.90	1.12	1.24	1.32	1.41	1.45	1.49

$$\begin{bmatrix} W_1 \\ W_2 \\ \vdots \\ W_k \end{bmatrix} = \begin{bmatrix} \omega_{1a} & \omega_{1b} & \omega_{1c} \\ \omega_{2a} & \omega_{2b} & \omega_{2c} \\ \vdots & \vdots & \vdots \\ \omega_{kn} & \omega_{kb} & \omega_{kc} \end{bmatrix} [m_a, m_b, m_c]. \tag{15}$$

As can be seen from the above formula, $\max(W_1, W_2, \ldots, W_k)$ will be the most preferred offload server solution.

6. Channel Selection Stage

At this level, we present the channel selection stage strategy. After completing the edge server selection, it is necessary to solve how to select the appropriate wireless channel for computation offloading, assuming that there is a set of available wireless channels $M = \{1, 2, \ldots, m\}$ between the edge user and the edge server node. Then, the user decision is $a_{n,m}$. Multiple users simultaneously selecting the same wireless channel for beneficial computation offloading can cause severe signal interference.

Defining the total cost of user n as Z_n, the objective function of the multiuser multichannel computing offload decision problem is defined as

$$\min_{a_{n,m} \in \{0,1\}} \sum_{n \in N} Z_n. \tag{16}$$

Using the potential game [38] analyses the multiuser multichannel distributed computation offloading decision problem, the potential function is constructed to prove that the multiuser multichannel distributed computation offloading decision problem satisfies the potential game condition, and there is a Nash equilibrium point. Among them, the Nash equilibrium point [38] is defined as the optimal solution to the NP-hard [39] problem of the multiuser multichannel distributed computing offload.

6.1. Game Analysis of Multiuser Multichannel Distributed Computation Offloading. In the distributed computing offload decision, the set of computation offloading decisions for all users except the edge-side user n is $a_{N-n} = (a_{1,m}, \ldots, a_{n-1,m}, a_{n+1,m}, \ldots, a_{N,m})$, $\forall m \in M$. Whether the user n chooses local computing or edge server-side computing to reduce their overall overhead, i.e., is given as

$$\min_{a_{n,m} \in \{0,1\}} Z_n\left(a_{n,m}, a_{N-n}\right), \quad \forall n \in N, m \in M. \tag{17}$$

According to equations (4) and (8), the mathematical expression of the overall cost of user n can be derived as

$$Z_n\left(a_{n,m}, a_{N-n}\right) = \begin{cases} K_n^l, & a_{n,m} = 0, \\ K_n^c(a_N), & a_{n,m} = 1. \end{cases} \tag{18}$$

Then, we express the above problem as a strategic game $\Gamma_{\mathrm{MCC}} = (N, \{a_{n,m}\}_{mN,m,M,M}, \{Z_n\}_{neN})$; the edge user strategy set conforming to Nash Equilibrium is $a_N = (a_{1,m}, a_{2,m}, \ldots, a_{n,m})$. Then, according to the definition of Nash Equilibrium, if the multiuser system is in equilibrium, no user can change the strategy unilaterally to further reduce the overhead. i.e.,

$$Z_n\left(a_{n,m}, a_{N-n}\right) \le Z_n\left(a_{n,m}^*, a_{N-n}\right), a_{n,m} \in \{0,1\}, \quad n \in N, m \in M. \tag{19}$$

In this formula, $a_{n,m}^*$ denotes the decision after the edge user changes.

6.2. Proof of the Existence of Multiuser Multichannel Distributed Computation Offloading Nash Equilibrium Points. The potential game is a subset of the strategy game. Each subject will continually approach the optimal objective function after a finite iteration to find the optimal solution of the objective function, and each potential game obeys a potential function. Here, we need to construct a potential function to prove that the target problem is a potential game problem, and then there is a Nash equilibrium point. From (4), (8), and (10), we can get $K_n^c(a_N) \le K_n^l$ equivalent to

$$\begin{aligned} &\Rightarrow \lambda_n^t\left(t_{n,\mathrm{up}}^c(a_N) + t_{n,\mathrm{exe}}^c\right) + \lambda_n^e e_{n,\mathrm{up}}^c(a_N) \le \lambda_n^t t_n^l + \lambda_n^e e_n^l \\ &\Rightarrow \lambda_n^t\left(\frac{B_n}{C_n(a_N)} + t_{n,\mathrm{exe}}^c\right) + \lambda_n^e\left(\frac{q_n B_n}{C_n(a_N)} + T_n\right) \le \lambda_n^t t_n^l + \lambda_n^e e_n^l \\ &\Rightarrow \frac{B_n(\lambda_n^t + \lambda_n^e q_n)}{C_n(a_N)} + \lambda_n^t t_{n,\mathrm{exe}}^c + \lambda_n^e T_n \le \lambda_n^t t_n^l + \lambda_n^e e_n^l \\ &\Rightarrow C_n(a_N) \ge \frac{B_n(\lambda_n^t + \lambda_n^e q_n)}{\lambda_n^t t_n^l + \lambda_n^e e_n^l - \lambda_n^t t_{n,\mathrm{exe}}^c - \lambda_n^e T_n}. \end{aligned} \tag{20}$$

According to (2), we then have that the interference μ_n in the wireless channel has an extremum TH_n when the edge-side user n implements the beneficial computation offloading:

$$\mu_n = \sum_{i \in \{N\}: a_i = a_n} q_i g_i \le \frac{q_n g_n}{\left(B_n(\lambda_n^t + \lambda_n^e q_n)/2^{K\left(\lambda_n^t t_n^l + \lambda_n^* e_n^- - \lambda_n^t r_{n,\mathrm{sec}} - \lambda_n^e T_n\right)}\right) - 1} - \sigma, \tag{21}$$

$$\mathrm{TH}_n = \frac{q_n g_n}{\left(B_n(\lambda_n^t + \lambda_n^e q_n)/2^{K\left(\lambda_n^t t_n^l + \lambda_n^e e_n^l - \lambda_n^t t_{n,\mathrm{exe}}^c - \lambda_n^e T_n\right)}\right) - 1} - \sigma. \tag{22}$$

It can be seen that when the wireless channel interference is sufficiently low, it is beneficial for the user to adopt the edge server-side computation mode. Otherwise, the user should perform the computation task locally. Based on the above results, we can know that channel interference has an extreme value, which satisfies the potential game condition. And, the following potential function is constructed to prove that the multiuser

multichannel computation offloading satisfies the potential game:

$$\phi(a_N) = \frac{1}{2}\sum_{i=1}^{N}\sum_{j\neq i} q_i g_i q_j g_j I_{\{a_i=a_j\}} I_{\{a_i=1\}} + \sum_{i=1}^{N} q_i g_i \mathrm{TH}_i I_{\{q=0\}}. \tag{23}$$

The potential game has a finite increment property (FIP) [38]; its incremental path length is limited, and the game subject can reach the Nash equilibrium after a finite number of iterations. Any edge user updates its current decision $a_{k,m}, m \in M$ for $a^*_{k,\tilde{m}}, \tilde{m} \in M, k \in N$; when in Nash equilibrium, it will lead to an increase in overall overhead, i.e.,

$$Z_k\left(a_{k,m}, a_{N-k}\right) < Z_k\left(a^*_{k,\tilde{m}}, a_{N-k}\right), a_k \in \{0,1\}, \quad k \in N, m, \tilde{m} \in M. \tag{24}$$

The objective function is proved to be a potential function in three cases.

Case 1 ($a_{k,m} = 1, a^*_{k,\tilde{m}} = 1$). The user overhead $Z_k(a_{k,m}, a_{N-k})$ is inversely proportional to data uplink rate $C_k(a_N)$, according to (2), that is, proportional to channel interference. This implies that

$$\sum_{j\in N\setminus\{k\}:\, a_j=a_k} q_j g_j < \sum_{j\in N\setminus\{k\}:\, a_j=a^*_k} q_j g_j. \tag{25}$$

Since $a_{k,m} = 1, a^*_{k,\tilde{m}} = 1$, according to (22), (23), and (24), it is obvious that

$$\begin{aligned}
\phi\left(a_{k,m}\right) &< \phi\left(a^*_{k,\tilde{m}}\right) \\
&= \frac{1}{2}\sum_{k=1}^{N}\sum_{j\neq k} q_k g_k q_j g_j I_{\{a_j=a_k\}} - \frac{1}{2}\sum_{k=1}^{N}\sum_{j\neq k} q_k g_k q_j g_j I_{\{a_j=a^*_k\}} \\
&= \frac{1}{2} q_k g_k \sum_{j\neq k} q_j g_j I_{\{a_j=a_k\}} + \frac{1}{2} q_j g_j \sum_{j\neq k} q_k g_k I_{\{a_k=a_j\}} \\
&\quad - \frac{1}{2} q_k g_k \sum_{j\neq k} q_j g_j I_{\{a_j=a^*_k\}} - \frac{1}{2} q_j g_j \sum_{j\neq k} q_k g_k I_{\{a^*_k=a_j\}} \\
&= q_k g_k \sum_{j\neq k} q_j g_j I_{\{a_j=a_k\}} - q_k g_k \sum_{j\neq k} q_j g_j I_{\{a_j=a^*_k\}} < 0.
\end{aligned} \tag{26}$$

Case 2 ($a_{k,m} = 0, a^*_{k,\tilde{m}} = 1$). Since $a_{k,m} = 0, a^*_{k,\tilde{m}} = 1$, and $Z_k(a_{k,m}, a_{N-k}) < Z_k(a^*_{k,\tilde{m}}, a_{N-k})$, when the user changes his local computing decision to the mobile edge computing decision process, the overall overhead increases, that is, the channel interference is greater than the maximum extreme value of the beneficial mobile edge computing interference:

$$\sum_{j\in N\setminus\{k\}:\, a_j=a_k} q_j g_j > \mathrm{TH}_k, \tag{27}$$

$$\begin{aligned}
\phi\left(a_{k,m}\right) &< \phi\left(a^*_{k,\tilde{m}}\right) \\
&= \sum_{j=1}^{N} q_j g_j TH_k - \frac{1}{2}\sum_{k=1}^{N}\sum_{j\neq k} q_k g_k q_j g_j I_{\{a_j=a^*_k\}} \\
&= q_k g_k TH_k - \frac{1}{2} q_k g_k \sum_{j\neq k} q_j g_j I_{\{a_j=a^*_k\}} \\
&\quad - \frac{1}{2} q_j g_j \sum_{k\neq j} q_k g_k I_{\{a^*_k=a_j\}} \\
&= q_k g_k TH_k - q_k g_k \sum_{j\neq k} q_j g_j I_{\{a_j=a^*_k\}} < 0.
\end{aligned} \tag{28}$$

Case 3 ($a_{k,m} = 1, a^*_{k,\tilde{m}} = 0$). By a similar argument in the second case, since $a_{k,m} = 1, a^*_{k,\tilde{m}} = 0$, and $Z_k(a_{k,m}, a_{N-k}) < Z_k(a^*_{k,\tilde{m}}, a_{N-k})$, $\phi(a_{k,m}) < \phi(a^*_{k,\tilde{m}})$.

For the above three cases, both the definitions of the potential game are satisfied. Then, we can get $\phi(a_{k,m}) < \phi(a^*_{k,\tilde{m}})$ from $Z_k(a_{k,m}, a_{N-k}) < Z_k(a^*_{k,\tilde{m}}, a_{N-k})$.

Combined with the above proof, it can be known that the target problem (multiuser multichannel computation offloading problem in the Mobile Edge Cloud Service strategy) is a potential game problem, and there is a Nash equilibrium point. The Nash equilibrium point can be used as the optimal value for multiuser multichannel mobile edge computing task offloading.

6.3. Multiuser Multichannel Task Scheduling Algorithm

6.3.1. Algorithm Design. We design the multiuser multichannel task scheduling algorithm according to the finite increment property of the potential game and ensure that any asynchronous response update process reaches Nash equilibrium within a finite number of iterations. Algorithm 1 of the whole auction flow is described as follows:

Step 1: initialization
Step 2: all computing tasks are done locally, i.e. $a_n(0) = 0$
Step 3: **end initialization**
Step 4: **repeat** for each user n and server node in each decision slot t
Step 5: **transmit** the pilot signal on the chosen channel m to the mobile cloud server base-stations
Step 6: **receive** the information of the received powers on all channels from each mobile edge user n
Step 7: **compute** the best response set $\Delta_n(t)$ in the base-stations
Step 8: **if** $\Delta_n(t) \neq \phi$ **then**
Step 9: **send** RTU message to the cloud for contending for the decision update opportunity
Step 10: **if receive** the UP message from the cloud then
Step 11: **choose** the decision $a_{n,\tilde{m}}(t+1) \in \Delta_n(t)$ for next slot
Step 12: **else** choose the original decision $a_{n,m}(t+1) \in a_{n,m}(t)$ for next slot
Step 13: **end if**
Step 14: **else choose** the original decision $a_{n,m}(t+1) \in a_{n,m}(t)$ for next slot
Step 15: **end if**
Step 16: **until** END message is received from the mobile cloud server base-stations

Algorithm 1: Multiuser multichannel task scheduling algorithm.

Specifically, the user synchronizes with the clock signal from the wireless base station, and the time slot used to update the computation offloading decision is called a decision period, and each decision period includes two phases:

Radio interference measurement phase: at this stage, we measure interference on different channels to select the appropriate channel for access. In the current decision slot, each edge node user who selects mobile edge computation offloading mode (i.e. $a_{n,m}(t) = 1$) will transmit the pilot signal on its selected channel m and then measure the total received power $\rho_m(a_N(t)) \triangleq \sum_{i \in N:\ a_i(t)=1} q_i g_i$ of each channel $m \in M$ on the radio base station. And, the power information received on all channels will be fed back to the edge node user. Therefore, each user n can grasp the interference on its channel $m \in M$ from other users as

$$\mu_n e\left(m, a_{-n}(t)\right) = \begin{cases} \rho_m\left(a_N(t)\right) - q_n g_n, & a_{n,m}(t) = 1, \\ \rho_m\left(a_N(t)\right), & a_{n,k}(t) = 1, k \neq m. \end{cases} \tag{29}$$

The interference received on the channel m currently selected by the edge user is equal to the measured total power minus the signal power. For other channels that do not transmit the pilot signal, the interference received is equal to the measured total power.

Offloading decision update phase: at this stage, we motivate the multiuser computing offload's finite incremental properties by having an edge node user perform a decision update. Based on interference information measured on different channels $\mu_n(m, a_{-n}(t)), m \in M$, each edge node user first calculates its best response update set as

$$\Delta_n(t) \triangleq \left\{ \begin{array}{l} \tilde{m}:\ \tilde{m} = arg \min\limits_{m \in M} \mu_n\left(m, a_{-n}(t)\right) \\ \mu_n\left(\tilde{m}, a_{-n}(t)\right) < \mu_n\left(m, a_{-n}(t)\right) \end{array} \right\}. \tag{30}$$

Then, in case $\Delta_n(t) \neq \phi$ (i.e., user n can improve its offload decision), user n will send a request-to-update (RTU) message to the edge server node to indicate that it wants to contend for the decision update opportunity. Otherwise, user n will not compete for updates in the next decision slot and keep their current offloading decisions unchanged (i.e. $a_{n,m}(t+1) = a_{n,m}(t)$). The edge server node will select the user with the highest priority from the user who has sent the RTU and send an update-permission (UP) command to update the decision $a_{n,\tilde{m}}(t+1) \in \Delta_n(t)$ in the next time slot. For users who do not receive the UP command, they will not update their decision in the next time slot (i.e. $a_{n,m}(t+1) = a_{n,m}(t)$).

6.3.2. Analysis of Convergence and Solvability. From the finite increment attribute (FIP) of the potential game, the algorithm will converge to the Nash equilibrium of the multiuser multichannel computation offloading game in a limited number of decision slots. In the simulation experiment, when the edge server does not receive any RTU message of the edge user in any decision time slot, that is, the game has reached the Nash equilibrium. Then, the edge server broadcasts the end message to all edge users, indicating that the updating process of the computation offloading decision is terminated. We analyze the convergence and solvability of the distributed computation offloading algorithm by calculating the extreme value of the number of required decision slots for the computation offloading algorithm.

In each decision slot, each edge user will execute steps 3–10 in Algorithm 1 in parallel. Since most operations involve only some basic arithmetic computations, the main part is to calculate the optimal response set in step 5, which involves the computation and sequence of m-channels measurement data, usually with the computational complexity of $O(mlogm)$. So, the computational complexity in each decision slot is $O(mlogm)$. Assuming that it requires C decision slots to terminate the algorithm, the total computational complexity of the distributed computation offloading algorithm is $O(Cmlogm)$. Let

$$\begin{cases} \mathrm{TH}_{\max} \triangleq \max\limits_{n\in N}\{\mathrm{TH}_n\}, \\ Q_n \triangleq q_n g_{n,m}, \\ Q_{\max} \triangleq \max\limits_{n\in N}\{Q_n\}, \\ Q_{\min} \triangleq \min\limits_{n\in N}\{Q_n\}, \end{cases} \tag{31}$$

where TH_n, q_n, and $g_{n,m}$ are the channel interference extremum, transmission power, and channel gain, respectively. Because we need C decision slots to converge, we have the following inference.

When TH_n and Q_n are nonnegative integers for any $n \in N$, the distributed computation offloading algorithm will terminate within at most $(Q_{\max}/2Q_{\min})N^2 + (TH_{\max}Q_{\max}/Q_{\min})N$ decision slot, i.e.,

$$C \le \frac{Q_{\max}}{2Q_{\min}}N^2 + \frac{TH_{\max}Q_{\max}}{Q_{\min}}N. \tag{32}$$

Proof. According to (23),

$$\begin{aligned} 0 \le \phi(a_N) &\le \frac{1}{2}\sum_{i=1}^{N}\sum_{j\neq i} Q_{\max}^2 + \sum_{i=1}^{N} Q_{\max} TH_{\max} \\ \Rightarrow 0 \le \phi(a_N) &\le \frac{1}{2}Q_{\max}^2 N^2 + Q_{\max} TH_{\max} N. \end{aligned} \tag{33}$$

In a decision slot, assume that an edge user $k \in N$ updates its current decision $a_{k,m}, m \in M$ to $a^*_{k,\tilde{m}}, \tilde{m} \in M$. And, this decision leads to a reduction in the overall cost of the user, i.e., $Z_k(a_{k,m}, a_{N-k}) > Z_k(a^*_{k,\tilde{m}}, a_{N-k})$. According to the definition of the potential function, it can be seen that the potential function will also be reduced by at least $Q_{\min}$, i.e.,

$$\phi_k\left(a_{k,m}\right) \ge \phi_k\left(a^*_{k,\tilde{m}}\right) + Q_{\min}. \tag{34}$$

We will consider three situations: (a) $a_{k,m} = 1, a^*_{k,\tilde{m}} = 1$; (b) $a_{k,m} = 0, a^*_{k,\tilde{m}} = 1$; (c) $a_{k,m} = 1, a^*_{k,\tilde{m}} = 0$.

While $a_{k,m} = 1, a^*_{k,\tilde{m}} = 1$, we can see from equation (26) that

$$\phi\left(a_{k,m}\right) - \phi\left(a^*_{k,\tilde{m}}\right) = Q_k\left(\sum_{j\neq k} Q_j I_{\{a_j = a_k\}} - \sum_{j\neq k} Q_j I_{\{a_j = a_k^*\}}\right) > 0. \tag{35}$$

Since Q_j are integers for any $j \in N$, then

$$\sum_{j\neq k} Q_j I_{\{a_j = a_k\}} \ge \sum_{j\neq k} Q_j I_{\{a_j = a_k^*\}} + 1. \tag{36}$$

According to the above formula,

$$\phi\left(a_{k,m}\right) \ge \phi\left(a^*_{k,\tilde{m}}\right) + Q_k \ge \phi\left(a^*_{k,\tilde{m}}\right) + Q_{\min}. \tag{37}$$

While $a_{k,m} = 0, a^*_{k,\tilde{m}} = 1$, we can get by formula (28) that

$$\phi\left(a_{k,m}\right) - \phi\left(a^*_{k,\tilde{m}}\right) = Q_k\left(TH_k - \sum_{j\neq k} Q_j I_{\{a_j = a_k^*\}}\right) > 0. \tag{38}$$

Since Q_k is the integer for any $j \in N$,

$$\mathrm{TH}_k \ge \sum_{j\neq k} Q_j I^I_{\left[a_j = a_k^*\right)} + 1. \tag{39}$$

According to the above formula,

$$\phi\left(a_{k,m}\right) \ge \phi\left(a^*_{k,\tilde{m}}\right) + Q_k \ge \phi\left(a^*_{k,\tilde{m}}\right) + Q_{\min}. \tag{40}$$

While $a_{k,m} = 1$ and $a^*_{k,\tilde{m}} = 0$, through a similar argument to the second case, we get

$$\phi\left(a_{k,m}\right) \ge \phi\left(a^*_{k,\tilde{m}}\right) + Q_k \ge \phi\left(a^*_{k,\tilde{m}}\right) + Q_{\min}. \tag{41}$$

Therefore, according to (33)–(41), we know that the algorithm will terminate by driving the potential function $\phi_k(a_N)$ to a minimum point within at most decision slots:

$$C \le \frac{\phi_{\max}(a_N)}{Q_{\min}} = \frac{Q_{\max}}{2Q_{\min}}N^2 + \frac{\mathrm{TH}_{\max}Q_{\max}}{Q_{\min}}N. \tag{42}$$

For the general case, the numerical results of the previous section indicate that the distributed computation offloading algorithm can also converge quickly, and the number of convergence decision slots increases linearly (almost) as the number of users N increases. The inferences in this section further indicate that the distributed computation offloading algorithm can converge quickly under normal conditions and has a quadratic convergence time $C_{\max}$ (i.e., an upper bound). Note that in the simulation experiment, the transmission power and channel gain are nonnegative (i.e., $q_n, g_{n,m} \ge 0$), so we know that $Q_n = \{q_n g_{n,m}\} \ge 0$. The nonnegative condition $\mathrm{TH}_n \ge 0$ ensures that each user has the opportunity to implement a beneficial computing offload (otherwise, the user can only choose the local computing all the time). Therefore, the algorithm makes sense only when Q_n and TH_n are nonnegative integers. $\mathrm{TH}_{\max}$, $Q_{\max}$, and $Q_{\min}$ can be obtained from known conditions, and then the algorithm can be solved. □

7. Simulation and Analysis

7.1. Experiment Settings. In this experiment, the face recognition algorithm [40] has been used as a computation task. MATLAB is used to simulate the computation offloading framework proposed in this paper. We will set up 5 edge server nodes in each experimental scenario, including 5 wireless base stations and 5 edge servers, running 30 virtual machines on each edge server, and the computing power of each virtual machine is set to 10GH [40]. The coverage of the base station is 100 meters [27], and edge users are randomly distributed in coverage [27]. Each task is executed by a single virtual machine. The various data parameters [3, 33, 41–44] in the simulation experiment are shown in Table 3.

7.2. Analysis of Simulation Results. In this section, we analyze the simulation results and discuss the performance of the framework proposed.

Table 3: Various data parameter tables in the simulation experiment.

Parameter	Values	Parameter	Values
M	5	λ_n^t	0.5
k	5	λ_n^e	0.5
$k\prime$	5	β	0~100 m
B_n	5 MB	α	4
D_n	1000 M cycles	g_n	$\beta^{-\alpha}$
f_n^l	1 GHz	q_n	100 mW
f_n^v	10 GHz	σ	−100 dBm
f_n^c	200~300 GHz	K	5 MHz
γ_n^l	1 mW		

7.2.1. Performance Analysis of Beneficial Computation Offloading Decision Strategy. We consider the two indicators of beneficial computation offloading users and real-time system overhead. The two indicators of the beneficial computation offloading decision strategy used in this paper are evaluated. Experiments are set up for analysis: (1) the change of the number of beneficial computation offloading users with the change of the decision time slot among the 50 users who perform the computation offloading is measured; (2) the change of the real-time system overhead of the user with the change of the decision time slot among the 50 users who perform the computation offloading is measured; (3) the comparison of the number of beneficial computation offloading users with the change of the user number of different computation offloading decision strategies (random computation offloading decision, beneficial computation offloading decision, and full computation offloading decision). All test data in this paper are the average of 100 trials.

From Figure 3, we can see the dynamic process of the number of beneficial computation offloading users under the beneficial computation offloading decision strategy. It shows that the strategy can increase the number of beneficial computation offloading users in the system and converge to a balance. From Figure 4, we can see the dynamic process of the overall cost of the mobile device user system under the beneficial computing offload decision strategy. It shows that the strategy can also keep the total overhead of the mobile device user system in the process of computation offloading from decreasing and eventually converging to an equilibrium. Figure 5 shows that, under the condition that each edge service node has sufficient computing resources in the mobile edge network, the number of beneficial computation offloading users will increase with the number of users. The performance of the computation offloading decision strategy used in this paper compared with the other two strategies is improved by 30%.

7.2.2. Performance Analysis of Cov-AHP Based Server Selection Strategy. We focus on the stability evaluation of the Cov-AHP-based server selection strategy used in this paper. Two experiments are set up: (1) compare the server weight of different server selection strategies in the same region [45] with the change of the task execution time; (2) compare the server weight of the server selection strategy used in this paper with the change of the task execution time in different

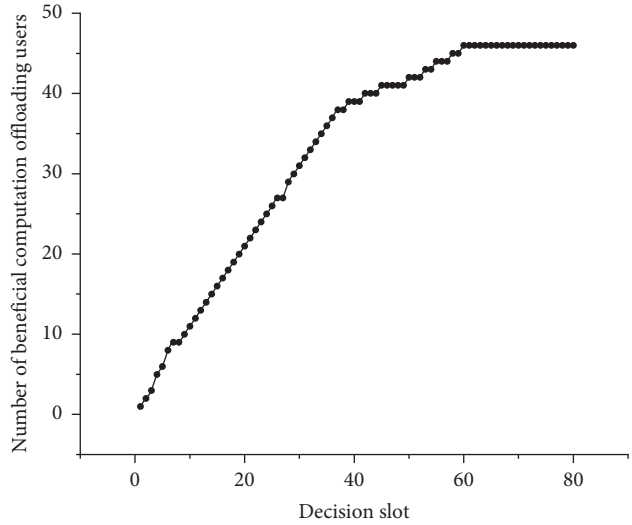


Figure 3: Dynamic beneficial computation offloading users.

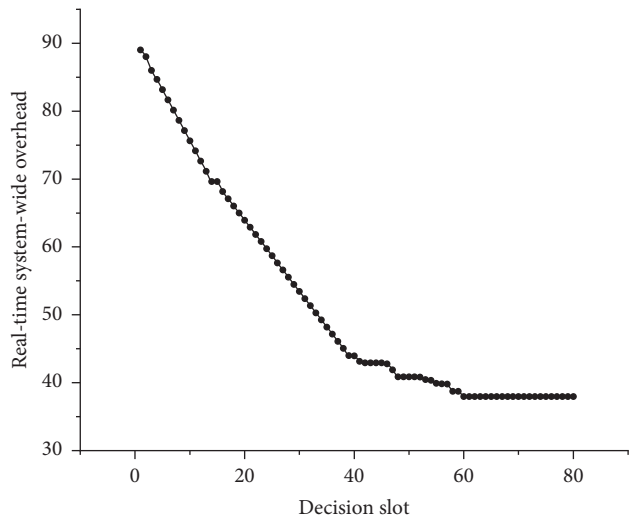


Figure 4: Real-time computing offload system overhead.

regions. In this experiment, we set the performance parameters of the server to be more realistic, and the performance of the server has advantages and disadvantages. The specific parameters are shown in Table 4.

The server weights based on different server selection strategies fluctuate with the task execution time in the same area (Figure 6). The server selection strategy based on Cov-AHP can reduce the weight fluctuation of each server in the same area effectively, and so that the user offloading decision is relatively stable. It also achieves load balancing between servers in the same area effectively.

The server weight based on the server selection strategy adopted in this paper varies with the task execution time in different regions (Figure 7). These regions depend on where the users are. The server selection strategy based on Cov-AHP can maintain the weight of each server in different regions effectively, so that the server weight fluctuation is small in the same region, which indicates that the server selection strategy based on Cov-AHP can efficiently achieve

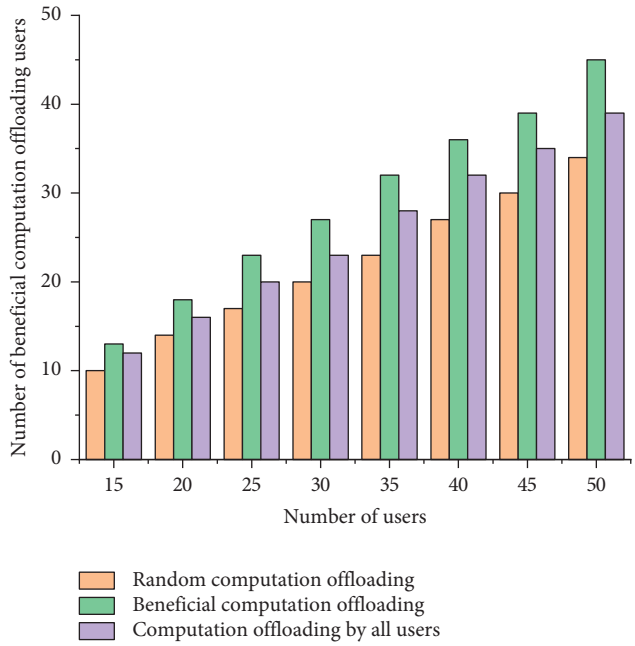


Figure 5: The number of beneficial computation offloading users for different computing offload decision strategies under different user numbers.

Table 4: Server performance parameters.

	Bandwidth (M)	Number of VM	VM CPU computing speed (GHz)	Server CPU computing speed (GHz)
Server 1	5	40	8	320
Server 2	6	30	10	300
Server 3	7	20	14	280
Server 4	4	30	10	300
Server 5	6	30	9	270

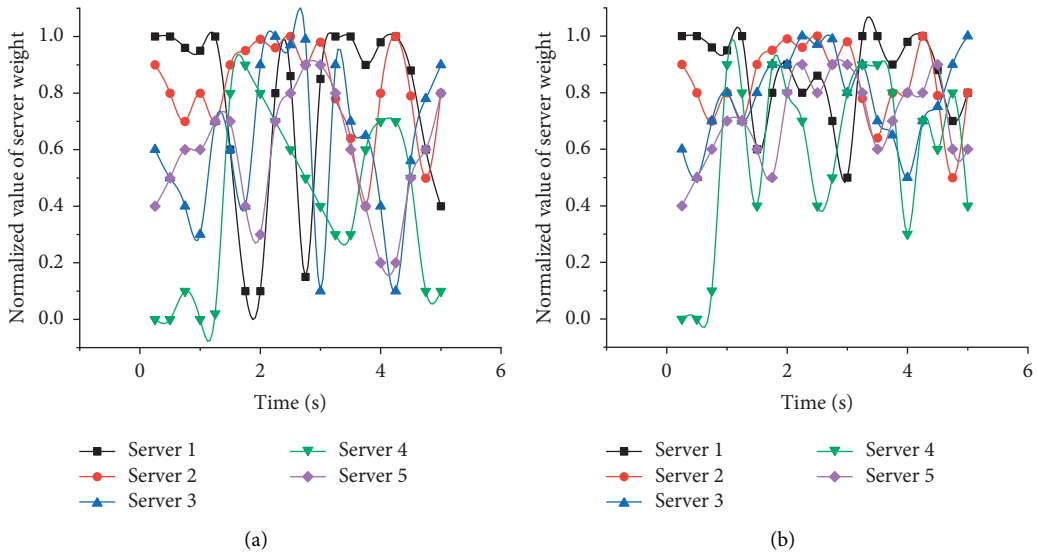
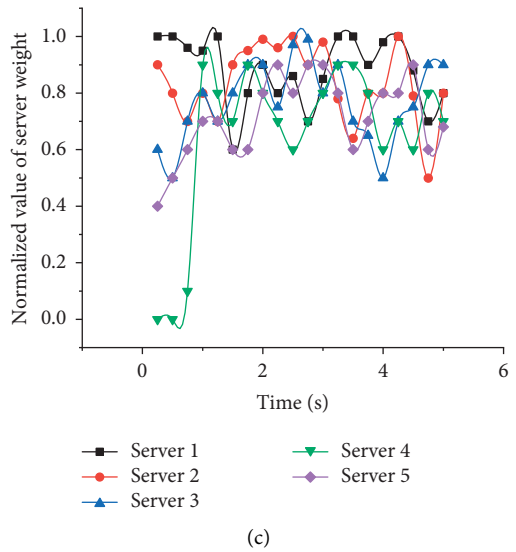
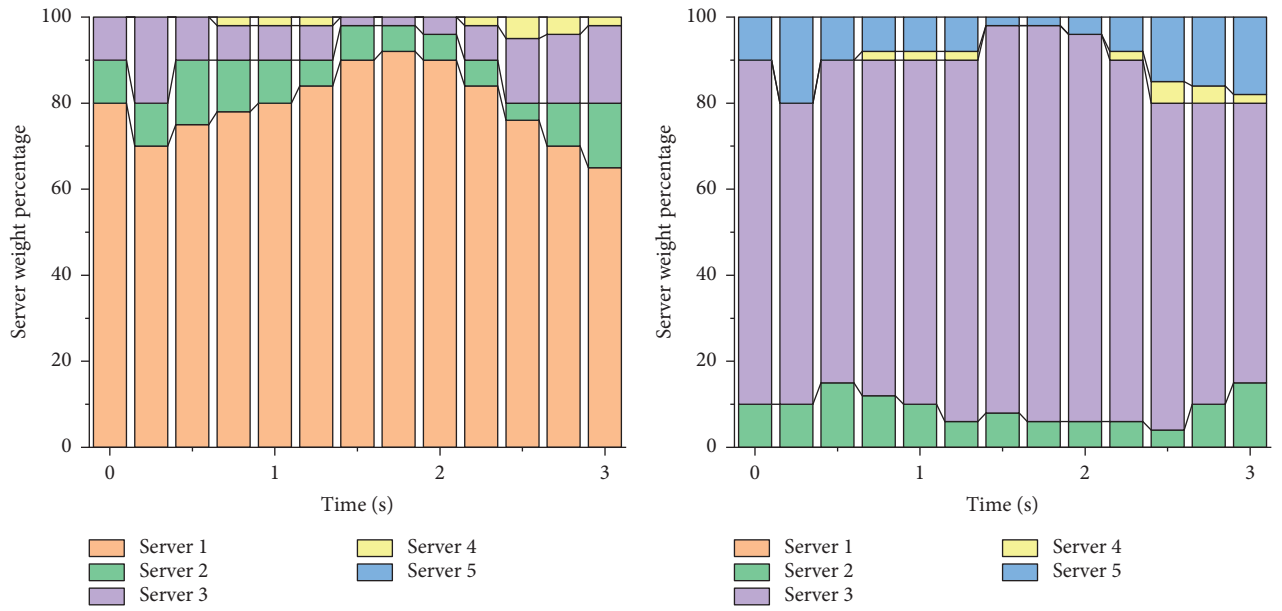


Figure 6: Continued.



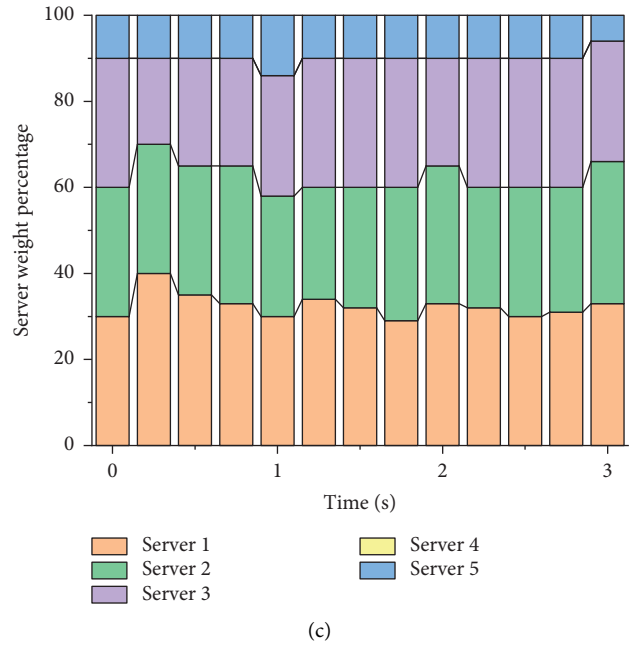
(c)

Figure 6: Server weight fluctuation based on (a) a random server selection strategy, (b) the polling server selection strategy, and (c) the Cov-AHP server selection strategy.



(a) (b)

Figure 7: Continued.



(c)

Figure 7: Server load fluctuations in different regions based on the Cov-AHP server selection strategy. (a) Region 1. (b) Region 2. (c) Region 3.

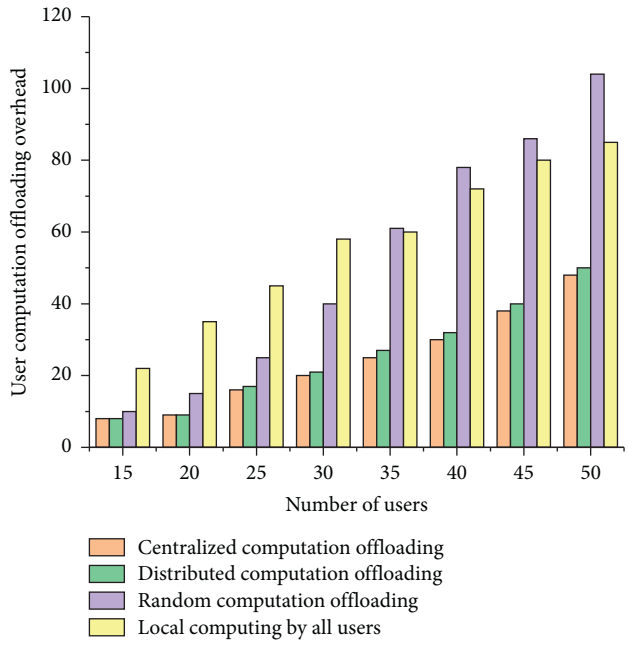


Figure 8: User overhead of different channel selection models.

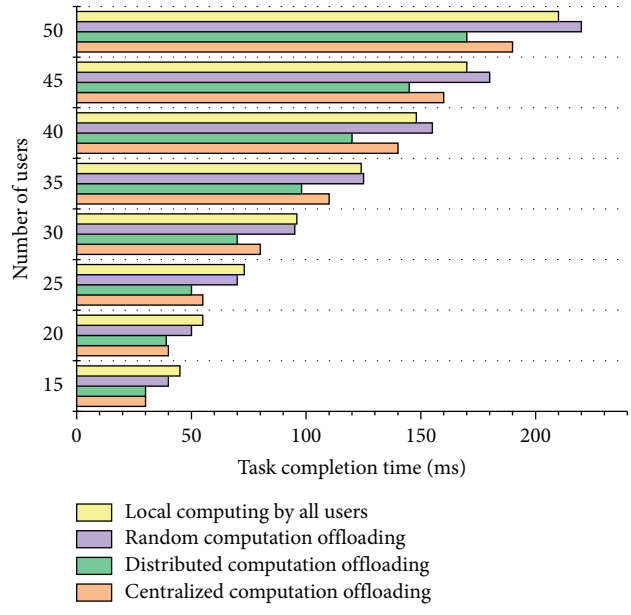


Figure 9: Task completion time of different channel selection models.

load balancing between servers in different areas within the coverage of the base station.

7.2.3. Performance Analysis of Multiuser Multichannel Distributed Computation Offloading Strategy Based on Potential Game. In this section, we focus on two important indicators of the user's computation offloading overhead and task completion time in the process of making channel selection decisions and completing the computation offloading. We evaluate two indicators of the multiuser multichannel distribution computation offloading strategies based on potential games is used in this paper. And, two experiments are set up separately: (1) compare the total user overhead of different channel selection strategies [2] in the process of computation offloading; (2) compare the time of different channel selection strategies [2] to complete the computation task. Among them, the centralized computation offloading algorithm uses the global optimization method to calculate

the overall cost of centralized computation offloading and requires the edge server to continuously interact with the edge users. It has been proved to be able to effectively find the approximate optimal solution of the complex combinatorial optimization problem.

The variation of the total user overhead of different channel selection strategies to complete the computation offloading process with the increase of the number of users is shown in Figure 8. The multiuser multichannel distributed computation offloading strategy based on the potential game has a larger advantage relative to the random computation offloading and the full local computing, which saves twice as much on average. The centralized computation offloading strategy is almost consistent or even slightly superior to the user overhead of the strategy in this paper, but it pays a large price in terms of delay. The task completion time of different channel selection strategies increases with the increase in the number of users (Figure 9). The multiuser multichannel distributed computation offloading strategy based on potential game used in this paper will greatly reduce the completion time of tasks compared with other strategies. This is because the strategy in this paper makes the most suitable decision for each user based on their own combined channel conditions. The centralized offloading decision needs to collect all users' information for centralized analysis, which will greatly increase the additional delay and affect the QoS of the users.

8. Conclusions

This paper proposes a multilevel computation offloading optimization framework suitable for multiuser multichannel multiserver scenarios in MEC to meet the needs of computing-intensive applications on mobile devices. From the perspective of edge users, delay and energy consumption are used as the basis for computation offloading decision, and then the concept of beneficial computation offloading is proposed. Through the Cov-AHP strategy for multiobjective decision-making, we make a comprehensive evaluation of the optional edge server and select the appropriate server to perform computation offloading. It is proved that the multiuser multichannel distributed computation offloading accords with the potential game condition, and it indicates that there is always a Nash equilibrium point in the potential game. Then, a distributed computation offloading algorithm is designed, which can realize Nash equilibrium. The simulation results show that the proposed offloading framework has higher stability than the similar methods, which can effectively reduce the delay and the overall cost of energy consumption of the edge client and improve the execution speed of the computation offloading and the standby time of the mobile device.

In future work, we will consider how edge devices optimize computation offloading in ad hoc networks, which will make it possible for users to share computing resources in more urgent situations.

Data Availability

No data were used to support this study.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was partially supported by the National Natural Science Foundation of China (Grant no. U1603261) and the Natural Science Foundation Program of Xinjiang Province (Grant no. 2016D01A080).

References

[1] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing—a key technology towards 5G," *ETSI White Paper*, vol. 11, pp. 1–16, 2015.

[2] C. Dong and W. Wen, "Joint optimization for task offloading in edge computing: an evolutionary game approach," *Sensors*, vol. 19, no. 4, p. 740, 2019.

[3] X. Chen, L. Jiao, W. Li, and X. Fu, "Efficient multi-user computation offloading for mobile-edge cloud computing," *IEEE/ACM Transactions on Networking*, vol. 24, no. 4, pp. 2795–2808, 2015.

[4] K. Ren, C. Wang, and Q. Wang, "Security challenges for the public cloud," *IEEE Internet Computing*, vol. 16, no. 3, pp. 69–73, 2012.

[5] L. Huang, X. Feng, L. Zhang, L. Qian, and Y. Wu, "Multi-server multi-user multi-task computation offloading for mobile edge computing networks," *Sensors*, vol. 19, no. 1, p. 1446, 2019.

[6] M. Satyanarayanan, P. Bahl, R. Caceres, and N. Davies, "The case for VM-based cloudlets in mobile computing," *IEEE Pervasive Computing*, vol. 8, no. 4, pp. 14–23, 2009.

[7] Y. Mao, J. Zhang, and K. B. Letaief, "Dynamic computation offloading for mobile-edge computing with energy harvesting devices," *IEEE Journal on Selected Areas in Communications*, vol. 34, no. 4, pp. 3590–3605, 2016.

[8] M. Kamoun, W. Labidi, and M. Sarkiss, "Joint resource allocation and offloading strategies in cloud enabled cellular networks," in *Proceedings of the IEEE International Conference on Communications (ICC)*, pp. 5529–5534, London, UK, 2015.

[9] M. H. Chen, B. Liang, and M. Dong, "A semidefinite relaxation approach to mobile cloud offloading with computing access point," in *Proceedings of the IEEE 16th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pp. 186–190, Stockholm, Sweden, June 2015.

[10] A. Sehati and M. Ghaderi, "Energy-delay tradeoff for request bundling on smartphones," in *Proceedings of the INFOCOM IEEE Conference on Computer Communications*, pp. 1–9, Atlanta, GA, USA, May 2017.

[11] M. Lopez-Martin, B. Carro, J. Lloret, S. Egea, and A. Sanchez-Esguevillas, "Deep learning model for multimedia quality of experience prediction based on network flow packets," *IEEE Communications Magazine*, vol. 56, no. 9, pp. 110–117, 2018.

[12] K. Zhang, Y. Mao, S. Leng et al., "Energy-efficient offloading for mobile edge computing in 5G heterogeneous networks," *IEEE Access*, vol. 4, no. 12, pp. 5896–5907, 2016.

[13] M. H. Chen, B. Liang, and M. Dong, "Joint offloading and resource allocation for computation and communication in mobile cloud with computing access point," in *Proceedings of*

the IEEE INFOCOM IEEE Conference on Computer Communications, pp. 1–9, Atlanta, GA, USA, May 2017.

[14] Y. Wang, Q. He, D. Ye, and Y. Yang, "Formulating criticality-based cost-effective fault tolerance strategies for multi-tenant service-based systems," *IEEE Transactions on Software Engineering*, vol. 44, no. 3, pp. 291–307, 2018.

[15] T. Q. Dinh, J. Tang, Q. D. La, and T. Q. Quek, "Offloading in mobile edge computing: task allocation and computational frequency scaling," *IEEE Transactions on Communications*, vol. 65, no. 2, pp. 3571–3584, 2017.

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

[17] T. Q. Dinh, Q. D. La, T. Q. Quek, and H. Shin, "Learning for computation offloading in mobile edge computing," *IEEE Transactions on Communications*, vol. 66, no. 4, pp. 6353–6367, 2018.

[18] X. L. Xu, X. Zhang, X. H. Liu, J. Jiang, L. Qi, and M. Z. A. Bhuiyan, "Adaptive computation offloading with edge for 5G-envisioned Internet of connected vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 1, no. 1, pp. 1–10, 2020.

[19] X. L. Xu, C. X. He, Z. Y. Xu, L. Qi, S. Wan, and M. Z. A. Bhuiyan, "Joint optimization of offloading utility and privacy for edge computing enabled IoT," *IEEE Internet of Things Journal*, vol. 7, no. 4, pp. 2622–2629, 2019.

[20] X. L. Xu, X. Y. Zhang, H. H. Gao, Y. Xue, L. Qi, and W. Dou, "BeCome: blockchain-enabled computation offloading for IoT in mobile edge computing," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 6, pp. 4187–4195, 2019.

[21] P. Lai, Q. He, G. Cui, X. Y. Xia et al., "Edge user allocation with dynamic quality of service," in *Proceedings of the 17th International Conference on Service-Oriented Computing (ICSOC2019)*, Toulouse, France, October 2019.

[22] K. Cheng, Y. Teng, W. Sun, A. Liu, and X. Wang, "Energy-efficient joint offloading and wireless resource allocation strategy in multi-MEC server systems,," in *Proceedings of IEEE International Conference on Communications (ICC)*, pp. 1–6, Kansas City, MO, USA, May 2018.

[23] C. Shi, K. Habak, P. Pandurangan et al., "Cosmos: computation offloading as a service for mobile devices," in *Proceedings of the 15th ACM International Symposium on Mobile Ad Hoc Networking and Computing*, pp. 287–296, Philadelphia, PA, USA, August 2018.

[24] Q. He, G. Cui, X. Zhang et al., "A game-theoretical approach for user allocation in edge computing environment," *IEEE Transactions on Parallel and Distributed Systems*, vol. 31, no. 3, pp. 515–529, 2020.

[25] P. Lai, Q. He, M. Abdelrazek et al., "Optimal edge user allocation in edge computing with variable sized vector bin packing," in *Proceedings of the 16th International Conference on Service-Oriented Computing (ICSOC2018)*, pp. 230–245, Hangzhou, China, November 2018.

[26] J. Li, T. Cai, K. Deng et al., "Community-diversified influence maximization in social networks," *Information Systems*, vol. 92, pp. 1–12, 2020.

[27] Q. He, J. Han, F. Chen et al., "QoS-aware service selection for customisable multi-tenant service-based systems: maturity and approaches," in *Proceedings of the International Conference on Cloud Computing (CLOUD2015)*, pp. 237–244, New York, USA, July 2015.

[28] L. Qi, Q. He, F. Chen et al., "Finding all you need: web APIs recommendation in web of things through keywords search," *IEEE Transactions on Computational Social Systems*, vol. 6, no. 5, pp. 1063–1072, 2019.

[29] W. Zhong, X. Yin, X. Zhang et al., "Multi-dimensional quality-driven service recommendation with privacy-preservation in mobile edge environment," *Computer Communications*, vol. 157, pp. 116–123, 2020.

[30] Y. Chen, N. Zhang, Y. Zhang, X. Chen, W. Wu, and X. Shen, "Energy efficient dynamic offloading in mobile edge computing for internet of things," *IEEE Transactions on Cloud Computing*, vol. 109, no. 10, pp. 1–11, 2019.

[31] X. Xu, Y. Chen, X. Zhang, Q. Liu, X. Liu, and L. Qi, "A blockchain-based computation offloading method for edge computing in 5G networks," *Software: Practice and Experience*, pp. 1–18, 2019.

[32] P. J. Darwen, "Computationally intensive and noisy tasks: co-evolutionary learning and temporal difference learning on backgammon," in *Proceedings of the 2000 Congress on Evolutionary Computation*, vol. 2, pp. 872–879, La Jolla, CA, USA, July 2000.

[33] K. Liu, J. Peng, H. Li, X. Zhang, and W. Liu, "Multi-device task offloading with time-constraints for energy efficiency in mobile cloud computing," *Future Generation Computer Systems*, vol. 64, no. 3, pp. 1–14, 2016.

[34] E. Cuervo, A. Balasubramanian, D. K. Cho et al., "MAUI: making smartphones last longer with code offload," in *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services*, pp. 49–62, San Francisco, CA, USA, June, 2010.

[35] L. Yang, J. Cao, Y. Yuan, T. Li, A. Han, and A. Chan, "A framework for partitioning and execution of data stream applications in mobile cloud computing," *Acm Sigmetrics Performance Evaluation Review*, vol. 40, no. 5, pp. 23–32, 2013.

[36] Y. Wen, W. Zhang, and H. Luo, "Energy-optimal mobile application execution: taming resource-poor mobile devices with cloud clones," in *Proceedings of the IEEE INFOCOM*, pp. 2716–2720, Orlando, FL, USA, March 2012.

[37] Z. Xie, "Cov-AHP: An improvement of analytic hierarchy process method," *Journal of Quantitative & Technical Economics*, vol. 8, pp. 137–148, 2015.

[38] O. Candogan, A. Ozdaglar, and P. A. Parrilo, "Dynamics in near-potential games," *Games and Economic Behavior*, vol. 82, no. 4, pp. 66–90, 2013.

[39] K. H. Loh, B. Golden, and E. Wasil, "Solving the maximum mardinality bin packing problem with a weight annealing-based algorithm," in *Operations Research and Cyber-Infrastructure*, pp. 147–164, Springer Science & Business Media, Berlin, Germany, 2009.

[40] T. Soyata, R. Muraleedharan, C. Funai, M. Kwon, and W. Heinzelman, "Cloud-vision: real-time face recognition using a mobile-cloudlet-cloud acceleration architecture," in *Proceedings of the IEEE Symposium on Computers and Communications (ISCC)*, pp. 000059–000066, Cappadocia, Turkey, July 2012.

[41] T. S. Rappaport, *Wireless Communications: Principles and Practice*, Prentice Hall PTR, Upper Saddle River, NJ, USA, 1996.

[42] M. Xiao, N. B. Shroff, and E. K. P. Chong, "A utility-based power-control scheme in wireless cellular systems," *IEEE/ACM Transactions on Networking*, vol. 11, no. 2, pp. 210–221, 2003.

[43] M. Chiang, P. Hande, T. Lan, and C. W. Tan, "Power control in wireless cellular networks power control in wireless cellular

networks mung chiang," *Foundations & Trends in Networking*, vol. 2, pp. 381–533, 2008.

[44] J. Wallenius, P. C. Fishburn, S. Zionts et al., "Multiple criteria decision making, multiattribute utility theory: recent accomplishments and what lies ahead," *Management Science*, vol. 54, no. 1, pp. 1336–1349, 2008.

[45] J. Sheng, J. Hu, X. Teng, B. Wang, and X. Pan, "Computation offloading strategy in mobile edge computing," *Information*, vol. 10, no. 4, p. 191, 2019.