Hindawi
Wireless Communications and Mobile Computing
Volume 2023, Article ID 5216943, 12 pages
https://doi.org/10.1155/2023/5216943

Research Article

Mobility-Aware Offloading and Resource Allocation Strategies in MEC Network Based on Game Theory

Changsen Xia,[1] **Zilong Jin**,[1,2] **Jian Su**,[1,2] **and Baozhu Li**[3]

[1]*School of Computer Science, Nanjing University of Information Science and Technology, Nanjing, China*
[2]*Jiangsu Collaborative Innovation Center of Atmospheric Environment and Equipment Technology (CICAEET), Nanjing University of Information Science and Technology, Nanjing 210044, China*
[3]*Internet of Things & Smart City Innovation Platform, Zhuhai Fudan Innovation Institute, Zhuhai, Guangdong, China*

Correspondence should be addressed to Baozhu Li; tiger_1984@yeah.net

Received 22 July 2022; Revised 29 August 2022; Accepted 24 November 2022; Published 1 February 2023

Academic Editor: Kalidoss Rajakani

With the emergence and development of communication technology and new computing paradigm named mobile edge computing (MEC), fast response and ultralow latency are given higher requirements. Nevertheless, due to the low penetration and coverage of the MEC network, it is difficult to guarantee the large-scale connection needs of all user groups in industry 4.0. In addition, user mobility is closely related to the network connection between edge nodes (ENs) and mobile devices (MDs) in industry 4.0, the frequent mobility of MDs makes the computation offloading process not smooth and the channel unstable, which can reduce the network performance. Hence, this paper constructs an edge network environment for MEC-based industrial internet of things (IIoT), considering the combined benefits of energy consumption, time delay, and computing resource cost to tackle the aforementioned problem by maximizing the utility of the entire system. In order to solve this problem, this paper proposes a mobility-aware offloading and resource allocation scheme (MAORAS). This scheme first employs the Lagrange multiplier method to solve the problem of computing resource allocation; then, a noncooperative game between MDs is established and the existence of Nash equilibrium (NE) has been proven. Simulation results demonstrate that the practical performance of the MAORAS optimization scheme could improve the system utility significantly.

1. Introduction

With the exponential growth of mobile applications data in IIoT, there is an urgent demand for real-time response, which makes it difficult for MDs with limited battery and computing resources to meet the requirements of low latency and real-time user experience for resource-intensive tasks [1, 2]. MEC emerged at the historic moment, sinking rich communication and computing resources to industrial devices, extending the computing power of MDs while also greatly reducing the burden on the backhaul link of the cloud computing centers [3, 4], which means that computation-intensive tasks, including object detection, face recognition, and model training, can be offloaded to MEC servers for further processing.

Nevertheless, MEC networks' penetration rate and coverage rate are relatively low, and it is difficult to guarantee the large number of connection needs of the dense user groups. The limited computing resources of the MEC server may increase the delay of task execution; therefore, it is critical to optimize computing resources and offloading mechanisms in order to decrease communication costs and increase system utility [5].

Multiuser resource allocation and offloading optimization in MEC have always been researching hotspots. In terms of optimization goals, it is mainly divided into three categories; one is to minimize time delay, the other is to reduce energy consumption simultaneously, and the third is to maximize system utility. With regard to solutions, there are online optimization algorithms based on Lyapunov, offloading strategies based on *Q*-learning, resource allocation solutions based on genetic algorithms, and so on.

Optimizing the offloading decisions and the corresponding resource allocation is a decision-making problem of

multiple participants essentially, i.e., MDs and MEC servers, and includes cloud centers sometimes, so game theory is widely utilized to make offloading decisions in the MEC scenario. One of the characteristics of MDs in edge computing network is mobility, which makes the network become a highly dynamic network and makes the optimization problem more complex. Generally, when a mobile user move from the original cell to other neighboring cell, handover will happen, contributing to the additional latency even packet loss, which will raise retransmission, and add the latency in turn. The current MEC studies based on game theory have the following shortcomings: (1) Most of the literature fails to consider the problem of task migration caused by user mobility; (2) single scenario, multiuser, and multiserver scenarios are not considered, and (3) game theory-based computation offloading and allocation schemes are mostly static allocations.

This paper researches the computation offloading and resource allocation problems in MEC networks under multiuser mobile scenarios to address the issues above. The main contributions of the article are as follows:

(i) The mobility of the user is characterized as the stay time of the user in the area. Also, the residence time of the user satisfies the exponential distribution. Comprehensively consider energy consumption, time delay, and computing cost, and then jointly optimize resource allocation and offloading decisions to maximize system utility

(ii) The optimization function of system utility maximization is formulated as an NP-hard problem and further divided into two subproblems, which are solved by Lagrangian multiplier method and game theory, respectively

(iii) The performance of mobility-aware offloading and resource allocation strategy (MAORAS) is evaluated by comparing it with other baseline algorithms. Simulation results show that the MAORAS could achieve better performance

This paper is organized as follows: Section 2 introduces the related work, whereas Section 3 presents the system framework and the optimization problem is formulated. In Section 4, the problem solution and algorithm proposal is presented. Experiment and simulation are performed in Section 5, and finally, the paper is summarized in Section 6.

2. Related Work

One of the most significant characteristics of MEC networks is that MDs move with the user's motion, rather than being stationary, contributing to the communication network being unstable. At the same time, frequent link disconnection makes user's task transfer fail, resulting in wasted energy consumption, increased latency, and reduced user satisfaction.

Existing research focused on resource allocation according to user mobility. Tang et al. proposed a DQN-based resource allocation scheme to solve the dynamic service prediction problem in a high-mobility environment [6]. For each EDC, a vector autoregressive model (VAR) was used to predict the future load to allocate computation resources. This method used the historical load time series of itself and its neighbors to predict the load, make decisions, and determine various management measures through the estimated resource usage of each EDC. On the premise of ensuring user service quality, we improve the utilization rate of EDC resources as much as possible. For problems such as how many resources to allocate in EDC [7], where to place different application services, and when to activate various resource management operations, a location-aware workload prediction method based on LSTM was proposed. The proposed method could predict the workload of the environmental data center in a short time by using the correlation of the workload changes of the environmental data center in the close-range environment. In order to solve the problem of road traffic congestion, Shinkuma et al. proposed a new predictive road traffic information delivery system architecture to ensure that the system can establish real-time delivery and prediction accuracy without overflowing the computational and network load [8]. Finally, the system was validated using numerical tests on real datasets and real network simulators, where the predicted road traffic information will be forwarded to users.

Existing machine learning methods have their own advantages for predicting user mobility [9]. Subramaniam and Kaur explored the operational efficiency of various machine learning methods [10], such as multiple perceptron, random forest, and support vector machine, running on mobile edge computing platforms. Ojima and Fujii evaluated the stability of the communication link between users and edge computing nodes through factors such as transmission power, path loss, shadow gain, standard deviation, and received power [11]. A Kalman filter is used to predict user mobility and select its nearest edge computing node to complete the delivery of the task results. Nasrin and Xie proposed a new handoff scheme for the problem of increased communication delay caused by user mobility [12]. In the study of mobility-based offloading, when a user leaves cellular coverage, MDs will send a report to the base station that contains relays, cell identities, and user speed from neighboring macro- and microbase stations. The user's priority is then assigned based on the three parameters: user speed, computing task delay requirements, and resource availability of mobile edge computing nodes. Finally, the base station decides whether to migrate to mobile edge computing nodes or remote cloud, based on resource availability.

However, the above research methods lack effective network environment awareness techniques for MEC networks. Also, they do not consider the changes in the offloading overhead and migration cost of user equipment as the location changes.

There are multiple users in the wireless communication network, the shared communication and computing resources are competitive, and game theory can effectively solve the decision-making problem of multiple participants

on the target so that each user can meet their own communication needs to the greatest extent. At the same time, the performance of the network is improved. The choice problem of computation offloading devices was studied in [13], a potential game for edge computing is constructed, in which each device selfishly reduces its payoff, study the computation offloading strategy optimization problem, and a game-based method developed to tackle such a complex problem. Reference [14] proposed a resource allocation scheme based on dynamic game to allocate resources variant devices, and the blockchain technology is utilized to record the whole resource transaction process to ensure confidentiality and anonymity.

In the scenario of NOMA-based MEC network, Wang et al. considered users and MEC servers as leaders and followers and then established a Stackelberg game [15]. The user minimizes task offloading and the total energy consumption of local computing by optimizing the task allocation coefficient and transmit power. The MEC server allocates computing resources to process offloading tasks, with the ultimate goal of minimizing execution delays. In the multiuser computation offloading problem of interference perception in MEC network [16], a computing offloading algorithm Nash-Q-learning based on Nash equilibrium and reinforcement learning is designed. The simulation results show that the algorithm can effectively reduce the overhead of the system under different system parameters. The performance of the MEC network is significantly impacted by the problem of frequency division and intra-area interference [17], which treated user computation offloading, CPU capacity adjustment, transmit power regulation, and network interference as a game. In this game, the utility of each mobile user is maximized, which has been proven that the game was an exact potential game with a Nash equilibrium. The simulation results show that the algorithm can effectively improve the performance of the MEC system.

3. System Architecture and Problem Formulation

3.1. System Architecture. Consider a multiuser and multibase station edge computing network system as illustrated in Figure 1. The scene consists of a macrobase station (MBS) and multiple small base stations (SBS). MEC server deployed on SBS, providing powerful computing capability for the SBS. Each user moves within the coverage of the base station. Define the set of users and edge servers as $N = \{1, 2\cdots,n\}$ and $M = \{1, 2,\cdots,m\}$. The computation offloading strategy for all users is expressed as $A = \{a_1, a_2\cdots,a_n\}$, $a_n = 1$ indicates that the task of user n is offloaded to the MEC server, and $a_n = 0$ means the task is processed locally. For each MD $\text{Task}_i = \{D_i, a_i, C_i, T_i^{\text{stay}}\}$, where D_i is the data size, a_i is the offloading decision variable, $C_i$is the computing resources required to complete the task, and T_i^{stay} is the time that the user stays in the area.

3.2. Local Computing Model. Equations should be provided in a text format, rather than as an image. Microsoft Word's equation tool is acceptable. Equations should be numbered consecutively, in round brackets, on the right-hand side of the page. They should be referred to as Equation (1), etc. in the main text.

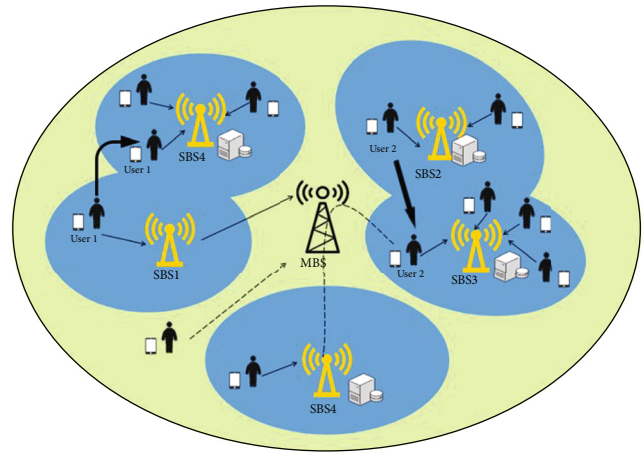

Figure 1: System model of MAORAS architecture.

When Task_i of the MD is executed locally, the local computing delay is expressed as

$$T_i^{\text{local}} = \frac{D_i \cdot C_i}{f_i^{\text{local}}}, \tag{1}$$

where C_i means the number of CPU cycles of unit data and f_i^{local} is the computing capacity of ith MD.

To evaluate the energy consumption generated by the MD in local processing, the energy consumption model $\varepsilon = \kappa f^2$ is used, where κ is the energy consumption coefficient depending on the chip structure. Therefore, when performing tasks locally, the energy consumption of ith MD is

$$E_i^{\text{local}} = \kappa\left(f_i^{\text{local}}\right)^2 D_i C_i. \tag{2}$$

3.3. MEC Computing Model. When the MD chooses to offload the task to the MEC server, the delay includes uplink transmission delay and MEC server processing delay. Since the size of the computing result is much smaller than the input data, the return delay of the computing result can be ignored. According to the Shannon formula, the uplink transmission rate can be calculated as

$$r^{\text{up}} = B_i \cdot \log_2\left(1 + \frac{p_i \cdot h_i}{I_i + \sigma^2}\right), \tag{3}$$

where B_i denotes channel bandwidth of base station, p_i is the transmit power of ith MD, and h_i means channel gain between ith MD and base station. I_i is the interference of other users in the channel, and σ^2 is the noise power.

The uplink transmission delay of the ith MD and the MEC server can be expressed as

$$T_i^{\text{up}} = \frac{D_i}{r^{\text{up}}}. \tag{4}$$

After the task is uploaded to the MEC server, the computing delay of the task processing can be expressed as

$$T_i^{\text{exe}} = \frac{D_i \cdot C_i}{f_i^{\text{mec}}}, \tag{5}$$

where f_i^{mec} represents computing resource allocated to the ith MD.

In summary, the total computation offloading delay of processing Task_i can be expressed as follows:

$$T_i^{\text{mec}} = T_i^{\text{up}} + T_i^{\text{exe}} = \frac{D_i}{r^{\text{up}}} + \frac{D_i \cdot C_i}{f_i^{\text{mec}}}. \tag{6}$$

In terms of energy consumption, only the energy consumption of the user in the transmission link is considered.

$$E_i^{\text{mec}} = p_i T_i^{\text{up}} = \frac{p_i \cdot D_i}{r^{\text{up}}}. \tag{7}$$

3.4. User Mobility Model. In the model of user mobility, we assume that the stay time of each user in the area obeys the following probability density function, denoted by $f_{T_i}(t)$, is expressed as

$$f_{T_i}(t) = \frac{1}{T_i} e^{t/T_i}, t \geq 0, \tag{8}$$

where T_i represents the average stay time of the ith MD, due to the heterogeneity of MDs, T_i is also different, and we assume that T_i obeys a Gaussian distribution.

3.5. Problem Formulation. In this section, a utility function is established that considers the three indicators of delay, energy consumption, and computing resource cost. We use the saved time T_i^{gain} and energy consumption E_i^{gain} as the benefit of the MD and use the computing resource cost and migration cost C_i^{mig} as the overhead. Benefit minus cost is the utility function of the MD.

$$\begin{aligned} T_i^{\text{gain}} &= T_i^{\text{local}} - T_i^{\text{mec}}, \\ E_i^{\text{gain}} &= E_i^{\text{local}} - E_i^{\text{mec}}. \end{aligned} \tag{9}$$

Due to the user's mobility, the stay time of user in the area will also affect the user's offloading decision and utility function. Therefore, the utility function is discussed in three different situations according to the user's stay time.

Case 1. The user stayed in the area for a short time and left the area before uploading the task to MEC servers, which can be expressed as $T_i^{\text{stay}} < T_i^{\text{up}}$. The probability of case 1 is $P_{T_i}(T_i^{\text{stay}} < T_i^{\text{up}})$. Therefore, the user cannot complete the transmission of the entire computing task, and the offloading strategy can only choose to local offloading, that is, $a_i = 0$ and utility function is

$$u_i^1 = 0. \tag{10}$$

Case 2. When the user completes the upload of the computing task and leaves the coverage before the MEC server execution is completed, which can be expressed as $T_i^{\text{up}} \leq T_i^{\text{stay}} < T_i^{\text{mec}}$, the probability of case 2 is $P_{T_i}(T_i^{\text{up}} \leq T_i^{\text{stay}} < T_i^{\text{mec}})$, then the user needs to pay an additional migration cost. Assuming that the cost of migration is only related to the size of data, given as $C_i^{\text{mig}} = \delta \cdot D_i$. So, the utility function in this case is

$$u_i^2 = \alpha \cdot T_i^{\text{gain}} + \beta \cdot E_i^{\text{gain}} - \gamma \cdot f_i^{\text{mec}} - \delta \cdot D_i, \tag{11}$$

where α and β are the coefficient of delay and energy consumption saving revenue and γ and δ are the coefficients of the migration cost for computing resource cost.

Case 3. After the MEC server task is accomplished, the user still stays in this area, which can be expressed as $T_i^{\text{stay}} \geq T_i^{\text{mec}}$, the probability of case 3 is $P_{T_i}(T_i^{\text{stay}} \geq T_i^{\text{mec}})$, and the utility function in this case is

$$u_i^3 = \alpha \cdot T_i^{\text{gain}} + \beta \cdot E_i^{\text{gain}} - \gamma \cdot f_i^{\text{mec}}. \tag{12}$$

Combine (10), (11), and (12), we can get the expression as follows:

$$u_i = \begin{cases} u_i^1, & P_1\left(T_i^{\text{stay}} < T_i^{\text{up}}\right), \\ u_i^2, & P_2\left(T_i^{\text{up}} \leq T_i^{\text{stay}} < T_i^{\text{mec}}\right), \forall i \in N, \\ u_i^3, & P_3\left(T_i^{\text{stay}} \geq T_i^{\text{mec}}\right). \end{cases} \tag{13}$$

The objective function of the system is defined as the sum of all user utility functions, expressed as

$$\begin{aligned} \max\ Z &= \sum_{i \in N, j \in M} u_i(A_{ij}, F_{ij}) \\ C_1 &: 0 \leq f_{ij}^{\text{mec}} \leq F \\ C_2 &: \sum_{i \in N, j \in M} f_{ij}^{\text{mec}} \leq F \\ s.t.\quad C_3 &: a_{ij} = \{0, 1\} \\ C_4 &: \sum_{i \in N, j \in M} a_{ij} \leq 1 \\ C_5 &: u_i \geq 0. \end{aligned} \tag{14}$$

Constraints C_1 and C_2 indicate that the computing resources allocated by the MEC server to users are

nonnegative, and the total computing resources allocated to users cannot exceed the maximum value. Constraints C_3 and C_4 indicate that the task can only be processed locally or offloaded to the MEC server, and a MD can only be offloaded to one MEC server. C_5 indicates that the user's utility function must be greater than or equal to 0, so that the revenue generated is greater than the local processing, which makes MEC offloading meaningful.

4. Proposed Scheme

4.1. Problem Decomposition. In this section, we propose MAORAS optimization scheme to solve the above optimization problem. The optimization problem is mainly divided into two parts. Where the offloading decision A is a binary variable, computing resource F is a continuous variable. So the optimization problem is a mixed integer nonlinear programming (MINLP) problem; thus, it is an NP-hard problem. According to the objective function and constraints, the constraints C_3, C_4, and C_5 of the offloading decision A are decoupled from the constraints C_1 and C_2 of the computing resource allocation F. The optimization problem is decomposed into two subproblems. Subproblem $P1$ is the problem of computing resource allocation under a given offloading decision.

$$\begin{aligned} &\mathrm{P1}: \max\ u_i(F) \\ &C_3: a_{ij} = \{0, 1\} \\ s.t.\ &C_4: \sum_{i\in N, j\in M} a_{ij} \le 1 \\ &C_5: u_i \ge 0. \end{aligned} \tag{15}$$

Subproblem $P2$ is the problem of offloading decision under a given computing resource allocation.

$$\begin{aligned} &\mathrm{P2}: \max\ u_i(A) \\ &C_1: 0 \le f_{ij}^{\mathrm{mec}} \le F \\ s.t.\ &C_2: \sum_{i\in N, j\in M} f_{ij}^{\mathrm{mec}} \le F. \end{aligned} \tag{16}$$

Subproblem $P1$ used the Lagrange multiplier method to solve the problem of computing resource allocation, and then, a noncooperative game is used to solve subproblem $P2$.

4.2. Computing Resource Allocation. According to Equations (13) and (14), the objective function can be expressed as follows:

$$\begin{aligned} \max\ Z = &\sum_{i\in N, j\in M} u_i^1 \cdot P_1\left(T_i^{\mathrm{stay}} < T_i^{\mathrm{up}}\right) \\ &+ u_i^2 \cdot P_2\left(T_i^{\mathrm{up}} \le T_i^{\mathrm{stay}} < T_i^{\mathrm{mec}}\right) \\ &+ u_i^3 \cdot P_3\left(T_i^{\mathrm{stay}} \ge T_i^{\mathrm{mec}}\right). \end{aligned} \tag{17}$$

Since the objective function is to maximize the benefit of the system, the local time delay and energy consumption are fixed values, so the objective function can be transformed into minimizing the total cost of the system, that is,

$$\begin{aligned} \min\ Q = &\sum_{i\in N, j\in M} (\alpha \cdot T_i^{\mathrm{mec}} + \beta \cdot E_i^{\mathrm{mec}} + \gamma \cdot f_i^{\mathrm{mec}}) \cdot (P_2 + P_3) \\ &+ \delta \cdot D_i \cdot P_2. \end{aligned} \tag{18}$$

The first derivative and the second derivative are obtained for the objective function Q, and we get

$$\begin{aligned} \frac{\partial Q}{\partial f_i^{\mathrm{mec}}} &= \left(-\alpha \cdot \frac{D_i \cdot C_i}{\left(f_i^{\mathrm{mec}}\right)^2} + \gamma\right) \cdot (P_2 + P_3), \\ \frac{\partial^2 Q}{\partial \left(f_i^{\mathrm{mec}}\right)^2} &= \alpha(P_2 + P_3) \cdot \frac{2 \cdot D_i \cdot C_i}{\left(f_i^{\mathrm{mec}}\right)^3} > 0. \end{aligned} \tag{19}$$

The second derivative of Q with respect to f_i^{mec} is always greater than 0, and the constraints C_1 and C_2 are both linear; therefore, problem (18) is a convex optimization problem. According to the KKT (Karush-Kuhn-Tucker) optimization conditions, the optimal solution can be obtained by the Lagrange multiplier method.

Formulate problem (18) as a partial Lagrangian function:

$$\begin{aligned} L\left(f_i^{\mathrm{mec}}, \eta\right) = &\sum_{i\in N, j\in M} (\alpha \cdot T_i^{\mathrm{mec}} + \beta \cdot E_i^{\mathrm{mec}} + \gamma \cdot f_i^{\mathrm{mec}}) \cdot (P_2 + P_3) \\ &+ \delta \cdot D_i \cdot P_2 + \eta \cdot \left(\sum_{i\in N, j\in M} f_i^{\mathrm{mec}} - F\right), \end{aligned} \tag{20}$$

where η is the Lagrangian multiplier related to the computational resource constraints, satisfying $\eta \ge 0$.

Then, get the first derivative of $L(f_i^{\mathrm{mec}}, \eta)$ with respect to f_i^{mec}, which can be expressed as

$$\frac{\partial L\left(f_i^{\mathrm{mec}}, \eta\right)}{\partial f_i^{\mathrm{mec}}} = \left(-\alpha \cdot \frac{D_i \cdot C_i}{\left(f_i^{\mathrm{mec}}\right)^2} + \gamma\right) \cdot (P_2 + P_3) + \eta. \tag{21}$$

When the first derivative takes 0, the optimal solution for computing resource allocation is obtained

$$f^{*}{}_i^{\mathrm{mec}} = \sqrt{\frac{\alpha \cdot D_i \cdot C_i \cdot (P_2 + P_3)}{\gamma \cdot (P_2 + P_3) + \eta^*}}, \tag{22}$$

where η^* is the Lagrange multiplier, which can be updated by gradient descent.

$$\eta(t+1) = \left[\eta(t) - \pi(t)\left(\sum_{i\in N, j\in M} f_i^{\mathrm{mec}} - F\right)\right], \tag{23}$$

1: Initialization: number of iterations $t = 0$, maximum tolerance value ϖ
2: Calculate f_i^{mec} according to formula (22).
3: Update Lagrange multipliers η according to formula (23).
4: If $\eta(t+1) - \eta(t) \le \varpi$.
5: then algorithm convergence, return $f_i^{\text{mec}} = f_i^{\text{mec}}$
6: else
7: $t = t + 1$
8: end if
9: Repeat steps 1-8 until the algorithm converges or $t = t_{\max}$

Algorithm 1: MEC computing resource allocation optimization algorithm.

where t is the number of iterations, and $\pi(t)$ is the step size. The condition for convergence is

$$\eta(t+1) - \eta(t) \le \varpi, \tag{24}$$

where ϖ indicates the maximum tolerance value.

Therefore, the resource allocation algorithm in this section is shown in Algorithm 1.

4.3. Computation Offloading Strategy. In the previous section, the Lagrange multiplier method was adopted to solve the problem of computing resources allocation. In this section, a noncooperative game model is adopted to solve the problem of task offloading strategies. Define the game as $G = \{N, A, U\}$, where N is number of devices participating in the game, A is offloading strategy, and U is utility function. $a_{-i} = (a_1, \cdots, a_{i-1}, a_{i+1}, \cdots, a_N)$ represents the set of offloading strategies for all users except user i. $U(a_i, a_{-i}) = u_i$ represents the utility function of user i. When none of the participants improves their utility function by unilaterally changing their offloading strategy, then all participants can converge to a Nash equilibrium state.

Definition 1. For offloading strategy A, if $\forall i \in N$, has $U(a_i^*, a_{-i}^*) \ge U(a_i, a_{-i}^*)$, then the offloading strategy a_i^* is called the Nash equilibrium of game G.

Definition 2. If there is a potential function $\phi(a)$ in game G, when the user's offloading strategy unilaterally changes from a_i to a_i', and $a_i' \in A$. The following formula can be obtained $U(a_i, a_{-i}) - U(a_i', a_{-i}) = \phi(a_i, a_{-i}) - \phi(a_i', a_{-i})$, then the game G is an exact potential game.

Definition 3. If game G is an exact potential game and the strategy set is finite, then game G has finite increasing properties, so there must be a Nash equilibrium.

Argument 1: Game $G = \{N, A, U\}$ is an exact potential game, and the potential function $\phi(a)$ is as follows: $\phi(a) = (1 - a_i) \cdot u_1 + a_i \cdot u_{2,3}$, where $u_{2,3}$ represents the two cases of u_2 and u_3, due to they are all benefits generated by MEC processing, all can be combined for discussion.

Proof. When user i updates the offloading strategy a_i to a_i', a potential function $\phi(a) = (1 - a_i) \cdot u_1 + a_i \cdot u_{2,3}$ that satisfies the condition. Consider the following two cases: (1) $a_i = 0$, $a_i{}' = 1$; (2) $a_i = 1$, $a_i{}' = 0$.

(1) The offloading strategy of user i is changed from local processing to MEC processing

$$\begin{aligned}\phi(a_i, a_{-i}) - \phi\left(a_i{}', a_{-i}\right) &= u_1 - u_{2,3} \\ &= U(a_i, a_{-i}) - U\left(a_i{}', a_{-i}\right)\end{aligned} \tag{25}$$

(2) The offloading strategy of user i is changed from MEC processing to local processing

$$\begin{aligned}\phi(a_i, a_{-i}) - \phi\left(a_i{}', a_{-i}\right) &= u_{2,3} - u_1 \\ &= U(a_i, a_{-i}) - U\left(a_i{}', a_{-i}\right)\end{aligned} \tag{26}$$

To sum up, there is a potential function $\phi(a)$ for $G = \{N, A, U\}$, according to Definition 3, for every potential game with a finite strategy space exists a Nash equilibrium, which means that any player can change its offloading strategy in a finite time to reach a Nash equilibrium state. □

4.4. MAORAS Algorithm Scheme. Combining the aforementioned computing resource allocation algorithm and computation offloading strategy in Section 3.3, the MAORAS algorithm scheme is proposed in Algorithm 2. Each mobile user makes the best offloading decision according to the current state and reaches the Nash equilibrium state after multiple iterations of convergence, maximizing the system utility.

The MAORAS algorithm scheme mainly consists of two parts:

(1) Offloading strategy preprocessing: corresponding to lines 1-17 in the pseudocode. In this process, the offloading strategy of all mobile users is first initialized to local processing; then, according to the user's stay time in the area, it is divided into three levels; finally, an initial offloading strategy is obtained

1: Initialization:
2: Mobile user set $N = \{1, 2\cdots, n\}$, MEC server set $M = \{1, 2, \cdots, m\}$, offloading strategy $A = \{a_1, a_2\cdots, a_n\}$, utility function $U = \{u_1, u_2\cdots, u_n\}$, game model $G = \{N, A, U\}$
3: End Initialization
4: Input: $\alpha, \beta, \gamma, P_1, P_2, P_3$
5: For each user i, $i \in N$ do
6: Calculate $T_i^{\text{local}}, E_i^{\text{local}}, T_i^{\text{mec}}, E_i^{\text{mec}}$.
7: Calculate ${}^{*} f_i^{\text{mec}}$ by Lagrange multiplier method.
8: if $T_i^{\text{stay}} < T_i^{\text{up}}$, then update $a_i = 0, u_i^1 = 0$.
9: else
10: if $T_i^{\text{up}} \le T_i^{\text{stay}} < T_i^{\text{mec}}$,
11: update $a_i = 1, u_i^2 = \alpha \cdot T_i^{\text{gain}} + \beta \cdot E_i^{\text{gain}} - \gamma \cdot f_i^{\text{mec}} - \delta \cdot D_i$.
12: else
13: update $a_i = 1, u_i^3 = \alpha \cdot T_i^{\text{gain}} + \beta \cdot E_i^{\text{gain}} - \gamma \cdot f_i^{\text{mec}}$.
14: end if
15: end if
16: End for
17: Output: Initialized offloading strategy A_0.
18: Initialization: iteration $g = 0$
19: $g \longleftarrow g + 1$
20: While $i \le N$ do
21: calculate $U(a_i, a_{-i})$;
22: if $a_i(g) = \arg \max U(a_i, a_{-i}), \forall i \in N$
23: then update $a_i(g) = a_i^*$.
24: else $a_i(g) = a_i(g-1)$
25: End While
26: Output: Optimal computing resource allocation ${}^{*} f_i^{\text{mec}}$, and Nash equilibrium offloading strategy A^*.

Algorithm 2: MAORAS algorithm scheme.

(2) Game theory-based offloading strategy update: corresponding to lines 18-26 in the pseudocode. Let each mobile user traverse the set of offloading strategies and take the one with the largest utility function as the best response strategy in this iteration process, which can be expressed as

$$a_i(g) = \arg \max U(a_i, a_{-i}), \forall i \in N, \quad (27)$$

when the following formula is satisfied:

$$a_i(g) = a_i(g-1), \forall i \in N \quad (28)$$

After g iterations, there is no mobile users want to change their offloading strategy individually to improve their utility function and record the final set of Nash equilibrium offloading strategies

$$A^* = \{a_1^*, a_2^*, \cdots, a_N^*\}. \quad (29)$$

5. Simulation Results and Analysis

In this section, we set the environmental parameters of the simulation experiment and evaluate the performance of MAORAS algorithm scheme by comparing other baseline algorithms. The specific details are shown as follows.

Table 1: Simulation parameters.

Parameters	Value
Transmission bandwidth	10 MHz
MD computing capability	0.5 ~ 0.8 GHz
MEC computing capability	40 GHz
MD transmission power	0.8 W
Gaussian white noise power	3×10^{-13} W
Energy consumption factor κ	5×10^{-27}
Data size of task	100~150 MB
The number of CPU cycles of unit data cycles/MB	0.4 ~ 0.5 G
The price of time gain α	1 units/s
The price of energy gain β	1 units/J
The price of computing resource γ	2 units/GHz
The price of migration cost δ	0.2 units/MB
Probability density function P_1, P_2, P_3	0.1,0.3,0.6

5.1. Simulation Environment. The experiment uses Python3.7 and MATLAB R2018b simulation tools, the operating system is window10, the CPU processor used is Intel(R) Core(TM) i5-9500 CPU @ 3.00 GHz, and the memory size is 16 GB.

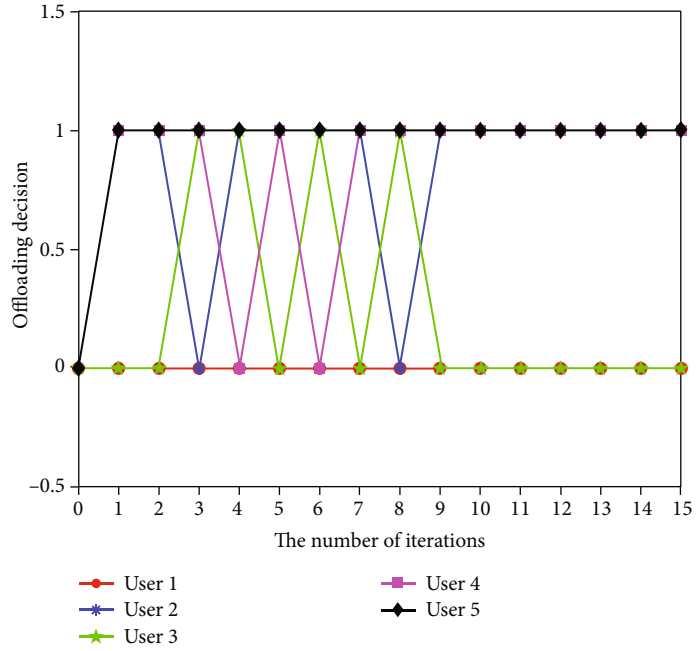

Figure 2: Changes in offloading decisions for different users.

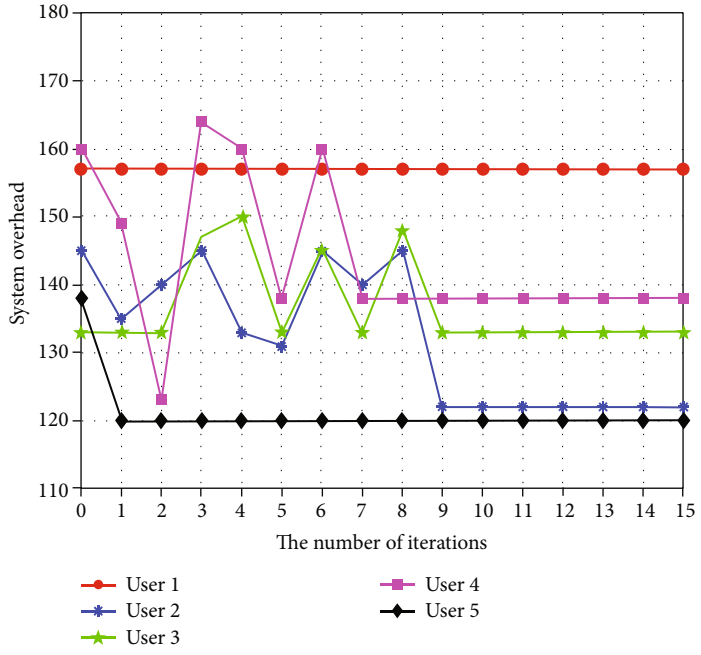

Figure 3: System overhead.

5.2. Parameter Settings. The scenario assumed in this article is on a square area. The users holding computing-demanding devices move randomly around the area. The task of the MDs can be executed locally or offloaded to the MEC servers. Due to the heterogeneity of MDs, assume that the number of mobile users is randomly selected from 5 to 50. Each region is equipped with a separate MEC server. The remaining parameters are shown in Table 1.

5.3. Algorithm Convergence. Figure 2 analyzes the convergence of mobile users' offloading decisions. When $N = 20$, randomly select 5 mobile users and observe the changes in their offloading strategies. The abscissa is the number of algorithm iterations, and the ordinate is the offloading decision.

In Figure 2, after a limited number of iterations, the Nash equilibrium strategy solution is finally obtained and

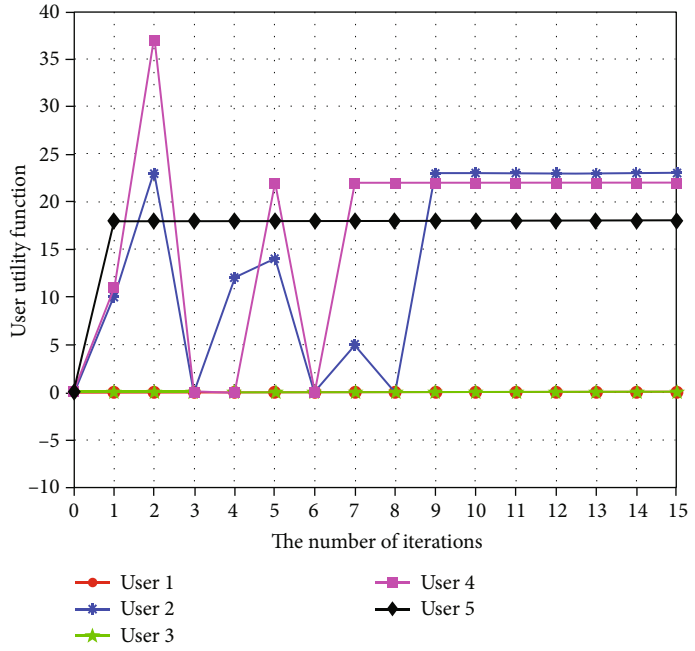

Figure 4: User utility function.

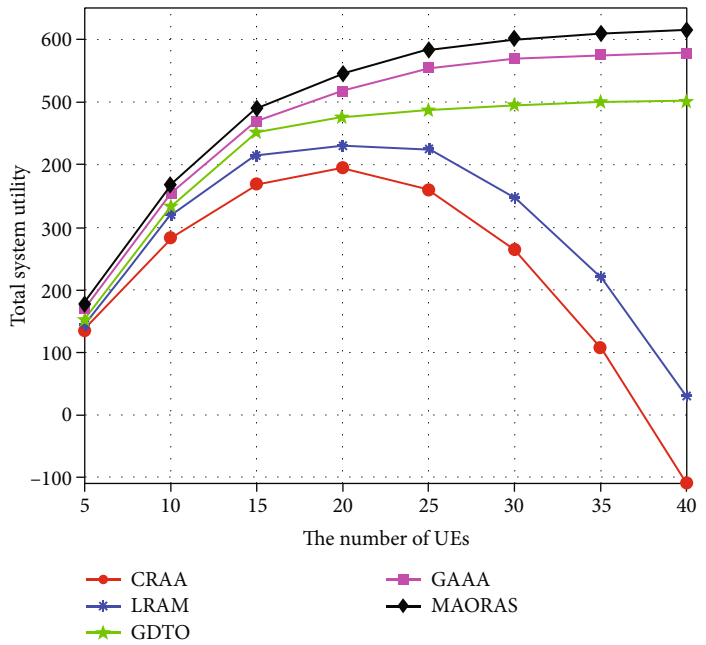

Figure 5: The effect of the number of users on system utility.

reaches a state of convergence. Initially, all MDs execute computation task locally. User 1 always insists on executing computing tasks locally, and User 2 and User 3 perform MEC offloading and local offloading, respectively, after 9 iterations. User 4 reaches the Nash equilibrium state after 7 iterations. User 5 insists on MEC offloading after only one strategy change. In general, after a finite number of iterations for all users, no one wants to unilaterally change strategy and finally reach a state of convergence.

Figures 3 and 4 show the changes in the system overhead and user utility function of five random users with the increase in the number of iterations. System overhead refers to the sum of delay, energy consumption, computing cost, and migration cost caused by user computation offloading. User utility function refers to the cost of local offloading minus MEC offloading. If $\text{cost}_i^{\text{local}} > \text{cost}_i^{\text{mec}}$, then it is more cost-effective to offload at MEC server, and the user utility function is also higher. Conversely, if $\text{cost}_i^{\text{local}} \leq \text{cost}_i^{\text{mec}}$,

choosing MEC offloading will cost more, on balance, the users prefer to handle it locally. Therefore, its utility function $u_i = 0$.

According to the change of the user's utility function, we can infer the user's attributes and make some reasonable guesses. For example, User 1 always insists on local decision-making. We speculate that he may be moving at high speed and stay in the area for a short time, which is not enough to upload the entire computing task before leaving. After many iterations of User 3, the cost of performing MEC offloading has always been higher than that of local processing. We can infer that its device has better performance and stronger computing power. For User 5, only one iteration insists on processing in MEC; it can be speculated that MEC provides powerful computing resources to User 5.

5.4. Algorithm Performance. The performance of the proposed MAORAS algorithm scheme is compared with the other algorithms. (1) Computing resource average allocation (CRAA): all users process tasks on the MEC server and each user gets the same computing resources. (2) Lagrange resource allocation method (LRAM): allocate computing resources according to the Lagrange multiplier method. (3) Game-based distributed task offloading (GDTO): game theory-based computing resource and offloading optimization scheme without considering user mobility. (4) Genetic algorithm-based allocation algorithm (GAAA): the user mobility model is the same as this paper but uses genetic algorithm.

Figure 5 presents the changes in the total system utility as the number of users increases in different algorithms. In the beginning, the utility function of the CRAA and LRAM algorithms increases with the number of users. When the number of users reaches 20, the system utility begins to show a downward trend; this is because the influx of a large number of users increases the load of the MEC server. The computing resources allocated to each user are less and less, and the transmission bandwidth of the system is occupied, causing network congestion. Meanwhile, if offload tasks to MEC servers randomly, the cost of MEC offloading is gradually higher than the cost of local offloading, and the final utility function is negative. Comparing CRAA and LRAM, it can be found that the computing resource allocation scheme using the Lagrange multiplier method is better than the traditional average allocation scheme.

Since GDTO does not consider user mobility, as the number of MDs increases, redundant devices can only be executed locally, and the number of tasks offloaded or migrated to the MEC server hardly changes. Hence, the GDTO algorithm tends to stabilize after the device reaches 25.

The total system utility of the GAAA and the MAORAS algorithm increases with the increase in the number of users. The system utility starts to change slowly but continues to increase and show a convergence trend. Because limited computing resources can only serve a part of computing tasks, the increase in the number of MDs provides the diversity and selectivity of offloading. It can be seen by observation that MAORAS is better than GAAA because the GAAA algorithm is prone to premature convergence.

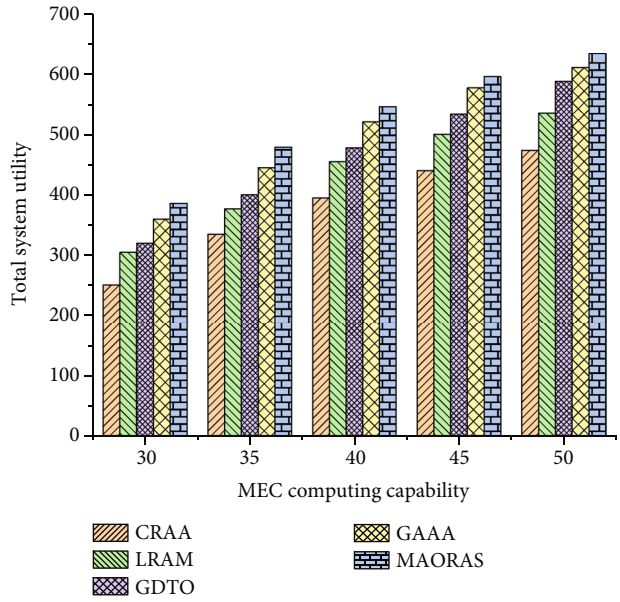

Figure 6: The effect of the MEC computing capability on system utility.

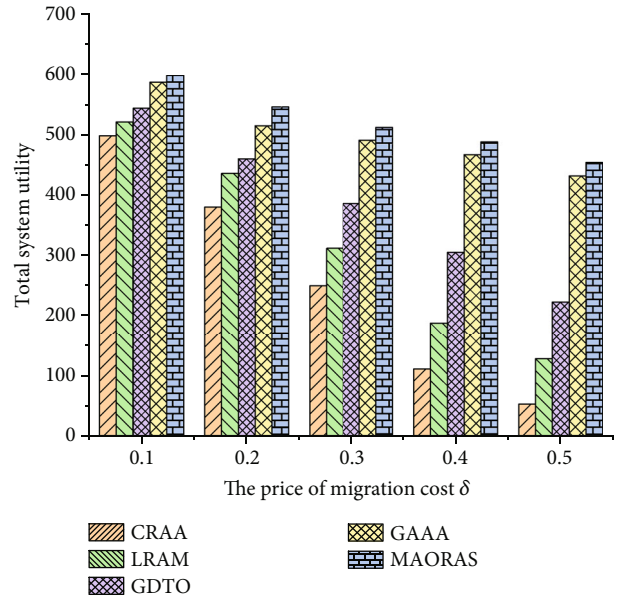

Figure 7: The effect of the migration cost on system utility.

Figure 6 shows the effect of the MEC computing capability on system utility using different algorithm, when $N = 20$, and the computing capability of the MEC server varies from 30 GHz to 50 GHz.

It shows that the five different algorithms' final convergent system utility value increases with the increase of computing capability. Compared with CRAA, the LRAM algorithm adopts the Lagrange multiplier method, which allocates computing resources more reasonably and improves the utility function of the system. The GDTO algorithm jointly optimizes the offloading decision and computing resource allocation strategy but does not

consider the user's mobility. Although the performance of the MEC server is improved, it is not considered to reduce the cost of migration. GAAA and MAORAS, considering the user's mobility, present good performance for those users who stay for a short time. More computing resources are provided to ensure that computation offloading is completed before the user leaves, thereby saving migration costs. MAORAS can improve by 5.3% compared with GAAA.

In Figure 7, the system utility of five different algorithms decreases with the increase in migration cost. The differences between the algorithms is initially small, but subsequently, the gap between them becomes larger and larger. The MAORAS algorithm proposed in this paper has the slowest decline rate, indicating that this scheme can significantly reduce the probability of migration. For CRAA and LRAM, all the tasks are offloaded or migrated to MEC server; the migration of many tasks resulted in a huge reduction in system revenue. Although GDTO took the offloading decision, it did not consider optimizing migration cost caused by user mobility. It can be concluded that ignoring user mobility will greatly increase the cost of migration and reduce system utility.

6. Conclusion

In this paper, a mobility-based edge computing scenario is considered in this paper, with the aim of maximizing system benefits by optimizing offloading decisions and resource allocation schemes. Using the Lagrange multiplier method to allocate computing resource, in the aspect of offloading decision, it is proved that the game model exists Nash equilibrium from establishing the model of the game relationship between the game participants, determining the final utility function of the game problem, and finally constructing the potential function. The simulation results show that the proposed MAORAS algorithm can effectively improve the utility function of the system. The performance of computation offloading is not only related to user mobility but also related to network status and routing. In the next step, we are prepared to predict the status of the edge computing network, including traffic prediction and so on, and conduct routing based on the prediction to reduce latency.

Data Availability

Data are available on request

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was sponsored by the National Natural Science Foundation of China under grant numbers 62271264, 61901191, 61972207, and 42175194 and the Shandong Provincial Natural Science Foundation (Grant No. ZR2020LZH005).

References

[1] B. Sonkoly, J. Czentye, M. Szalay, B. Nemeth, and L. Toka, "Survey on placement methods in the edge and beyond," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 4, pp. 2590–2629, 2021.

[2] Q. Luo, S. Hu, C. Li, G. Li, and W. Shi, "Resource scheduling in edge computing: a survey," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 4, pp. 2131–2165, 2021.

[3] M. Patel, B. Naughton, C. Chan, N. Sprecher, S. Abeta, and A. Neal, "Mobile-edge computing introductory technical white paper," *White Paper, Mobile-edge Computing (MEC) Industry Initiative*, vol. 29, pp. 854–864, 2014.

[4] Y. C. Hu, M. Patel, D. Sabella, N. Sprecher, and V. Young, "Mobile edge computing-a key technology towards 5G," *ETSI White Pater*, vol. 11, no. 11, pp. 1–16, 2015.

[5] Y. Wang, P. Lang, D. Tian et al., "A game-based computation offloading method in vehicular multiaccess edge computing networks," *IEEE Internet of Things Journal*, vol. 7, no. 6, pp. 4987–4996, 2020.

[6] F. Tang, Y. Zhou, and N. Kato, "Deep reinforcement learning for dynamic uplink/downlink resource allocation in high mobility 5G HetNet," *IEEE Journal on Selected Areas in Communications*, vol. 38, no. 12, pp. 2773–2782, 2020.

[7] C. Nguyen, C. Klein, and E. Elmroth, "Multivariate LSTM-based location-aware workload prediction for edge data centers," in *2019 19th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*, pp. 341–350, Larnaca, Cyprus, 2019.

[8] R. Shinkuma, S. Kato, M. Kanbayashi, Y. Ikeda, R. Kawahara, and T. Hayashi, "System design for predictive road-traffic information delivery using edge-cloud computing," in *2018 15th IEEE Annual Consumer Communications & Networking Conference (CCNC)*, pp. 1–6, Las Vegas, NV, USA, 2018.

[9] X. Wang, Y. Han, V. C. M. Leung, D. Niyato, X. Yan, and X. Chen, "Convergence of edge computing and deep learning: a comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 2, pp. 869–904, 2020.

[10] P. Subramaniam and M. Kaur, "Review of security in mobile edge computing with deep learning," in *2019 Advances in Science and Engineering Technology International Conferences (ASET)*, pp. 1–5, Dubai, United Arab Emirates, 2019.

[11] T. Ojima and T. Fujii, "Resource management for mobile edge computing using user mobility prediction," in *2018 International Conference on Information Networking (ICOIN)*, pp. 718–720, Chiang Mai, Thailand, 2018.

[12] W. Nasrin and J. Xie, "A joint handoff and offloading decision algorithm for mobile edge computing (MEC)," in *2019 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6, Waikoloa, HI, USA, 2019.

[13] Y. Ding, K. Li, C. Liu, and K. Li, "A potential game theoretic approach to computation offloading strategy optimization in end-edge-cloud computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 6, pp. 1503–1519, 2022.

[14] H. Xu, W. Huang, Y. Zhou, D. Yang, M. Li, and Z. Han, "Edge computing resource allocation for unmanned aerial vehicle assisted mobile network with blockchain applications," *IEEE Transactions on Wireless Communications*, vol. 20, no. 5, pp. 3107–3121, 2021.

[15] K. Wang, Z. Ding, D. K. C. So, and G. K. Karagiannidis, "Stackelberg game of energy consumption and latency in

MEC systems with NOMA," *IEEE Transactions on Communications*, vol. 69, no. 4, pp. 2191–2206, 2021.

[16] S. Liang, H. Wan, T. Qin, J. Li, and W. Chen, "Multi-user computation offloading for mobile edge computing: a deep reinforcement learning and game theory approach," in *2020 IEEE 20th International Conference on Communication Technology (ICCT)*, pp. 1534–1539, Nanning, China, 2020.

[17] N. Li, J. Yan, Z. Zhang, J. F. Martinez, and X. Yuan, "Game theory based joint task offloading and resources allocation algorithm for mobile edge computing," in *2020 16th International Conference on Mobility, Sensing and Networking (MSN)*, Tokyo, Japan, 2020.