J Inf Process Syst, Vol.17, No.3, pp.615~629, June 2021
https://doi.org/10.3745/JIPS.01.0078

ISSN 1976-913X (Print)
ISSN 2092-805X (Electronic)

JOURNAL OF INFORMATION PROCESSING SYSTEMS JIPS

Strategy for Task Offloading of Multi-user and Multi-server Based on Cost Optimization in Mobile Edge Computing Environment

Yanfei He* and Zhenhua Tang**

Abstract

With the development of mobile edge computing, how to utilize the computing power of edge computing to effectively and efficiently offload data and to compute offloading is of great research value. This paper studies the computation offloading problem of multi-user and multi-server in mobile edge computing. Firstly, in order to minimize system energy consumption, the problem is modeled by considering the joint optimization of the offloading strategy and the wireless and computing resource allocation in a multi-user and multi-server scenario. Additionally, this paper explores the computation offloading scheme to optimize the overall cost. As the centralized optimization method is an NP problem, the game method is used to achieve effective computation offloading in a distributed manner. The decision problem of distributed computation offloading between the mobile equipment is modeled as a multi-user computation offloading game. There is a Nash equilibrium in this game, and it can be achieved by a limited number of iterations. Then, we propose a distributed computation offloading algorithm, which first calculates offloading weights, and then distributedly iterates by the time slot to update the computation offloading decision. Finally, the algorithm is verified by simulation experiments. Simulation results show that our proposed algorithm can achieve the balance by a limited number of iterations. At the same time, the algorithm outperforms several other advanced computation offloading algorithms in terms of the number of users and overall overheads for beneficial decision-making.

Keywords

Cost Optimization, Distributed Computing, Game Theory, Mobile Edge Computing, Multi MEC Servers, Nash Equilibrium, Task Offloading

1. Introduction

The technology of smartphones, tablet computers and other mobile terminal equipment has promoted the emergence of high transmission and processing rate for many services and applications, bringing severe challenges to network service providers. Although smartphones' and other products' CPU, battery capacity, software and hardware are constantly upgraded, they are still limited by the physical design, and cannot process applications requiring large-scale computing in a short time. This poses a challenge and yet offers an opportunity to stimulate the cloud computing technology; thus allowing terminal users to access and use cloud computing [1-3]. In mobile cloud computing (MCC), the mobile device (MD)

Manuscript received May 18, 2020; first revision July 3, 2020; accepted July 24, 2020.

Corresponding Author: Yanfei He (zjyycit@163.com)
* Dept. of Information Technology, Zhejiang Yuying College of Vocational Technology, Hangzhou, China (zjyycit@163.com)
** Zhejiang Radio and Television Group Media Convergence Technology Center, Hangzhou, China (392791274@qq.com)

can gain access to the remote centralized cloud computing and storage resources by using the core network and the Internet of mobile operators. MCC can directly and effectively improve the data storage capacity of users. At the same time, by offloading the computing tasks of applications to the cloud for processing, the computing load is reduced and the energy consumption is saved, conducive to extending the battery life of user terminal equipment [4]. Moreover, sufficient computing resources can provide the premise for the design, use and promotion of complex applications.

However, due to the long transmission distance between the MCC central cloud and users, available wireless channels are limited, which causes great pressure on the wireless transmission and the backhaul load of mobile network. In addition, due to the overload of the service center, the computing task queuing will increase the delay cost and the service cost. Therefore, performance requirements of the delay-sensitive application cannot be met, and the quality of experience (QoE) cannot be guaranteed, resulting in huge profit loss to service providers.

In recent years, the number of wireless terminal devices is further increasing, and the demand for computing and terminal devices' data is expected to witness an exponential growth in the next few years. The growth mainly comes from new applications, which need low latency and low energy services with intensive computing or a large amount of data, for instance, tactile interactive services that require responses within millisecond delay, Internet of Things (IOT), machine type communications (MTC), large-scale online games, virtual reality, augmented reality, etc. To avoid the high latency, it is worth considering to locate the cloud service near the user; thus, the mobile edge computing (MEC).

In MEC [5,6], the intermediate computing resources are introduced between the MD and the core cloud data center, which will address the problem of insufficient computing power for terminal devices [7,8]. These edge computing resources can be used to minimize the execution delay of computing tasks within the coverage, and provide enough computing resources for data processing through task unloading [9,10]. Thus, in the context of MEC, it is very important to propose an effective strategy for data offloading and computing offloading.

The innovations of the proposed strategy are summarized as follows:

1) Our proposed strategy point outs the problems and deficiencies in the existing research. Furthermore, the task execution time cost model and task execution energy consumption model are defined in a complex MEC network scenario.
2) In the proposed strategy, the offloading weights are calculated, and the offloading decision is iteratively as well as distributedly updated. Thus, the decision-making of mobile equipment users can be done within local iterations, and the real-time computing requirement for all users can be met at the same time.

2. Related Research

Research on MEC mainly focuses on offloading strategies and the optimization of network resource allocation in different network scenarios. In the network scenario of the multi-user and the single MEC server, Mao et al. [11] proposed a joint optimization strategy, which can optimize the computation delay and equipment energy consumption at the same time. Wang et al. [12] further realized the joint optimization of wireless and computing resources based on the binary offloading strategy in the single

MEC server environment. The authors [13] solved the energy-efficient problem in the environment of the multi-user and the multi-MEC server. They divided the user-generated tasks into multiple sub-tasks and offloaded them in order to execute multiple computing nodes around.

Although some studies have paid attention to the problem of computing resource allocation in MEC systems, many early optimization methods and strategies have focused on the optimization of fixed computing resource allocation, and currently more is based on the new resource optimization technology DVFS for adjusting the CPU operating frequency [14]. Thinh et al. [15] studied a system scenario where a mobile equipment can offload computing tasks to multiple intervening nodes and can adjust its own CPU frequency, and proposed a joint optimization framework of offloading strategy and frequency adjustment. In the multi-user MEC environment, Zhang and Liu [16] proposed an offloading decision algorithm using the game theory, effectively minimizing the equipment's total energy consumption within the computing delay constraints. In order to reduce the energy consumption of the mobile equipment and to ensure users' task delay requirements, a task offloading decision algorithm based on the game theory was proposed. Tanzil et al. [17] minimized the execution delay of all tasks by allocating the computing resources of multiple MEC servers. Specifically, a service cluster was formed by cooperation. If the server performs a task for connecting other servers, it will offer this server a monetary reward. The server will first try to fulfil the task with the smallest communication delay. Only when the server is unable to process the task, will the task be transferred to other nearby servers for processing. In the scenario of a single mobile equipment with multiple access nodes, the task offloading strategy and mobile equipment frequency adjustment are jointly optimized by [18], which minimizes the total task execution delay and the total energy consumption.

However, many current studies on MEC ignore the characteristics of different tasks in a multi-user and multi-task environment, only a few works have noted this. Mao et al. [19] proposed a wireless resource management strategy for multi-user MEC system. Their strategy can minimize the weighted total energy consumption of the long-period wireless equipment. Huang et al. [20] discussed a multi-user oriented MEC task offloading algorithm using the deep Q-learning.

In summary, as a new network architecture paradigm, MEC helps to satisfy the growing demand for computing services and to achieve the collaborative optimization of network resources. It can reduce the system overheads and the task execution delay, and improve users' quality of service (QoS). In addition, the delay minimization, the energy minimization and the trade-off delay energy are generally considered. For the participants, they are divided into the single-user oriented and the multi-user oriented. However, further research is still needed to explore how to efficiently perform computation offloading based on the trade-off between the delay and the energy according to the characteristics of different tasks in a multi-user and multi-task scenario.

In order to address the above problems, this paper focuses on the optimization of offloading strategies, aimed at reducing the overall system overheads in complex MEC network scenarios, and studies how to formulate a reasonable computation offloading strategy to offload mobile nodes' own computing tasks to the edge server. We comprehensively consider various factors, for example, energy consumption, delay, cost, resource usage, contribution or profit of different types of tasks, and reduce the overall overheads of computation offloading. Furthermore, multi-users can efficiently offload computing in the game model. Therefore, in the environment of the multi-user and the multi-server MEC, this paper proposes a new task offloading algorithm considering the cost optimization.

3. Materials and Method

3.1 System Model

The overall system model based on the multi-user and the multi MEC server (MECs) is shown in Fig. 1. K MECs are represented as $\boldsymbol{K} = \{1,2,\dots,K\}$. The carrier of uplink communication is expressed as $\boldsymbol{N} = \{1,2,\dots,N\}$, and the bandwidth of each subcarrier is B_N. Multiple mobile terminals are represented as $\boldsymbol{I} = \{1,2,\dots,I\}$. Each mobile terminal has a task request. $A(D_i,\tau_i,X_i)$ represents the task attribute; D_i is the storage space occupied by the input data of the i-th task (unit: bit), τ_i is the calculation delay (unit: second), and X_i is the computing power required to execute the i-th task.

$f_{i,loc}$ is used to represent the local CPU dominant frequency of the i-th mobile terminal. The maximum data transmission power of all mobile terminals is equal to P^m. $f_{k,ser}$ is used to represent the CPU dominant frequency of the k-th MECs. $\boldsymbol{W} = \{w_{i,n,k} | w_{i,n,k} \in \{0,1\}, i \in I, n \in N, k \in K\}$ is used to represent the MECs allocation of subcarriers. Only when $w_{i,n,k} = 1$, the i-th mobile terminal can complete the task calculation on the k-th MECs with the help of subcarrier n. $\boldsymbol{P} = \{P_{i,n,k} | P_{i,n,k} \in [0,P^m], i \in I, n \in N, k \in K\}$ is used to represent the power allocation of the subcarriers, and $P_{i,n,k}$ is the wireless data transmission power of the subcarriers between the i-th mobile terminal and the k-th MECs. $\boldsymbol{G} = \{g_{i,n,k}, i \in I, n \in N, k \in K\}$ is used to represent all subcarrier channel gain sets, and $g_{i,n,k}$ represents the channel gain of subcarrier n between the i-th end user and the MECs. The noise of the proposed system obeys a Gaussian distribution δ^2 with an expected value of 0.

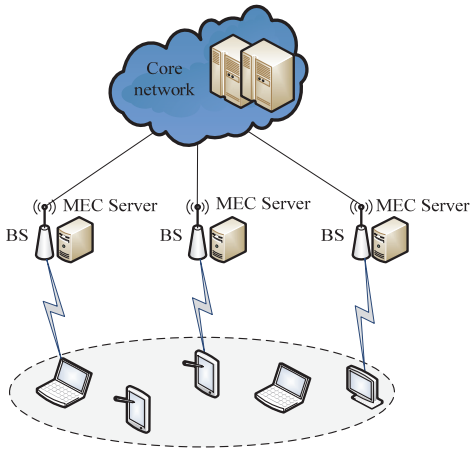

Fig. 1. The diagram of multi-user and multi-server system model.

3.2 Time Cost Model for Task Execution

Suppose that the total CPU cycles for completing the task is expressed as D_iX_i. Since the task has been divided into two execution modes (local and offload), the time overheads are also divided into two situations for specific discussion [21,22]:

Time overhead of local execution

The local execution time cost depends on users' own computing power $f_{i,loc}$, the local execution time

cost can thus be expressed as

$$T_i^l = \frac{D_i X_i}{f_{i,loc}} \tag{1}$$

Time overhead of remote offload execution

When users offload tasks, the task needs to be transferred to the MEC server, and then to be processed by the CPU in servers. Thus, for offloading tasks, the execution time cost is divided into two parts: transmission and calculation:

(1) Time overhead of transmission process: Based on the OFDMA's wireless transmission mechanism, due to the strict subcarrier allocation, the interference between users is ignored. The data transmission rate at which user i offloads the task to MEC server k can be expressed as:

$$R_{i,k} = B_N \sum_{n=1}^{N} w_{i,n,k} \log 2\left(1 + \frac{P_{i,n,k} g_{i,n,k}}{\delta^2}\right) \tag{2}$$

Consider that the data volume of computing tasks generated by users is relatively small, and the channel remains unchanged during the process of offloading tasks to MEC server. Therefore, the time overhead for user i to transfer tasks to MEC server is:

$$T_{i,k}^t = \frac{D_i}{R_{l,k}} \tag{3}$$

(2) Time overhead of calculation process: After receiving tasks offloaded by users, the MEC server will always allocate computing resources to execute tasks until the task is finally completed. In other words, only one computing task is executed at a time, and CPU resources will not be pre-empted by other tasks during the task execution. As only the idle MEC server will be used for the task scheduling and the execution, the queuing delay during the task offloading and the execution would not be considered. The time cost of executing task i in the MEC server depends on tasks' computational complexity and the CPU frequency, and can be expressed as

$$T_{l,k}^c = \frac{D_i X_i}{f_{k,ser}} \tag{4}$$

In sum, the total time cost of executing task i in the MEC server is

$$T_{i,k}^r = T_{i,k}^t + T_{i,k}^c = \frac{D_i}{R_{i,k}} + \frac{D_i X_i}{f_{k,ser}} \tag{5}$$

3.3 Energy Consumption Model for Task Execution

As the time-overhead model above, the energy consumption model is also separately discussed based on the mode of task execution.

Energy consumption of task local execution

Given the CPU frequency of tasks, the energy consumption of CPU per cycle can be expressed as

$k_0 f_{i,loc}^2$, where k_0 is a constant related to the user equipment CPU. Thus, the energy consumption of task *i* executed locally can be expressed as

$$E_i^l = k_0 f_{i,loc}^2 D_i X_i \tag{6}$$

Energy consumption of task offloading remote execution

When the task is offloaded to MEC server for execution, it also includes the energy consumption of two processes. One is the transmission energy consumption of offloading tasks from the user to MEC server, and the second is the execution energy consumption of the tasks in MEC server. Since the output result is often much smaller than the input data after the task is executed, we ignore the downlink transmission energy consumption when calculation results from MEC server are returned to the user. The transmission energy offloaded from task *i* to MEC server *k* can be expressed as

$$E_{i,k}^t = \sum_{n=1}^{N} w_{i,n,k} p_{i,n,k} \frac{D_i}{R_{i,k}} \tag{7}$$

Similar to the local execution of tasks, the calculated energy consumption of task *i* offloaded to MEC server *k* can be expressed as

$$E_{i,k}^c = k_1 f_{k,ser}^2 D_i X_i \tag{8}$$

where k_1 is a constant related to the *k* CPU of MEC server. In summary, the total energy consumption performed by *i* offloading to MEC server *k* can be expressed as

$$E_{i,k}^r = E_{i,k}^t + E_{i,k}^c = \sum_{n=1}^{N} w_{i,n,k} p_{i,n,k} \frac{D_i}{R_{i,k}} + k_1 f_{k,ser}^2 D_i X_i \tag{9}$$

3.4 Task Offloading Weight Model based on Shapley Value

In actual applications, there is a trade-off between energy consumption and offloading delay. It is difficult to assign different tasks with varied task offloading weights through a unified definition. Thus, in this study, the offloading weight is calculated by the offloading weight model based on the Shapley value, and the offloading weight is calculated fairly with the rewards of different tasks. Shapley value is the solution concept in the cooperative game theory. As a fair standard, it is often used to distribute profits among multiple participants [23,24]. For each cooperative game, the unique distribution scheme generated by the alliance of all participants can be obtained by the calculation of Shapley value [25].

Suppose there are *i* participants in the complete set $I = \{x_1, x_2, \dots, x_i\}$, and a subset $S \subseteq I$ formed by any number of people. $v(S)$ represents the value generated by the cooperation of elements included in S subset. Function v is a characteristic function. Then the final value $\psi_n(I, v)$ is Shapley value.

Shapley value guarantees the fairness of distribution and has the following four characteristics:

1) Effectiveness: All values are allocated, namely $\sum_{n \in I} \psi_n(I, v) - v(I)$.
2) Symmetry: If the positions of x_n and x_m are equivalent (i.e., they can be substituted for each other), the benefits of the two should be the same, namely $\psi_n(I, v) = \psi_m(I, v)$.
3) Pseudo-contribution: The income of non-contributors is 0.

4) Additivity: If the same batch of people completes two tasks, the benefits distribution of two tasks should be consistent with the results of separate distribution, i.e., $(v_1 + v_2)(s) = v_1(S) + v_2(S)$.

With proof, Shapley value is the only solution that satisfies the above four conditions [7,26]. Calculated by the following formula:

$$\psi_n(I,v) = \frac{1}{I!}\sum_{S\subseteq I\{n\}} |S|!\left(|I|-|S|-1\right)\left[v(S\cup n)-v(S)\right] \tag{10}$$

Shapley value is in fact the average value of marginal contribution. For offloading computation, by measuring the profit generated by the contribution of tasks, offloading decisions can be more fairly and effectively made.

In the calculation model, λ_i^t and λ_i^e are the tasks' weights for calculating delay and energy consumption respectively, and they satisfy $\lambda_i^t + \lambda_i^e = 1$. For mobile node *i*, its multiple tasks can calculate Shapley value of multiple tasks by the rewards generated. The total profit $v(I)$ is 1. Let $\lambda_i^t = \psi_n(I,v)$, then$\lambda_n^e = 1 - \lambda_i^t$. With this kind of weight calculation method, the task with larger contribution has a greater weight on the delay, and is dominant in the offloading.

4. Game-based Computation Offloading Strategy

By the computation offloading model obtained above, the task offloading strategy in the multi-user and multi-task scenario is solved; thereby, turning the problem into a multi-user game problem and offering a computation offloading strategy.

4.1 Multi-user Game Model

4.1.1 Problem of beneficial computation offloading

The lower the data transmission rate *i*, the greater the energy consumption of wireless access and the longer the transmission time. In this case, the mobile equipment *i* prefers local computing tasks to avoid high energy consumption and high latency caused by data offloading. The following is a beneficial computation offloading.

Definition 1. When the offloading decision a is fixed, compared with local computing, if the computing cost does not increase by using the edge server, the decision to choose mobile equipment *i* to offload to edge servers (i.e., $a_i > 0$) is beneficial (i.e., $K_i^c(a) \le K_i^m$).

Based on beneficial computation offloading, a centralized problem-optimization algorithm is firstly designed according to the overall performance indicators of the mobile equipment. The problem can be modeled as follows:

$$\begin{aligned} &\max \sum\nolimits_{i\in N} L_{\{a_i>0\}} \\ &s.t. K_i^c(a) \le K_i^m, \forall a_i > 0, i \in N \\ &\quad a_i \in \{0,1,\ldots,M\}, \forall i \in N \end{aligned} \tag{11}$$

where $L_{a_i>0}$ is the indicator function. If $a_i > 0$, then $L_{a_i>0} = 1$, otherwise $L_{a_i>0} = 0$.

However, the problem of finding the maximum number of the beneficial computation offloading equipment is an NP problem. This can be proved by being converted to the maximum cardinality binning problem. Given *N* objects, the size of object *n* is p_n. *M* boxes are given at the same time, with the same capacity *C*. The problem is that loading objects into boxes generates the largest number of objects and does not exceed the capacity limit of boxes. The problem can be expressed as:

$$\begin{aligned} &\max \sum_{n=1}^{N} \sum_{m=1}^{M} x_{n,m} \\ &s.t. \sum_{n=1}^{N} p_n x_{n,m} \le C, \forall m \in M, \\ &\quad \sum_{m=1}^{M} x_{n,m} \le 1, \forall n \in \boldsymbol{N}, \\ &x_{n,m} \in \{0,1\}, \forall n \in \boldsymbol{N}, m \in M \end{aligned} \tag{12}$$

The above mentioned maximum cardinality binning problem has proved to be an NP problem. If the mobile equipment is considered as the object of the maximum cardinal packing problem and the channel is the box from the maximum cardinal packing problem, the weight of objects can be expressed as $w_n = q_i g_{i,s}$. The capacity of the box can be expressed as the information rate of the channel. If mobile equipment *i* selects channel *m*, it can be assumed that the object *i* is packed in box *m*. The beneficial task offloading status can thus be expressed as the capacity limit of boxes. Therefore, the centralized optimization algorithm is also an NP problem.

4.1.2 Game model

Let $a_{-i} = (a_1, \ldots, a_{i-1}, a_{i+1}, \ldots, a_i)$ be the set of unloading decisions for all the equipment except equipment *i*. The equipment *i* will make its own unloading decisions according to a_{-i}. Choose local calculation (*i*) or offloading calculation by wireless channel ($a_i > 0$), and minimize its overall overheads by the following formula:

$$\min_{a_i \in A_i \triangleq \{0,1,\ldots,M\}} Z_i(a_1, a_{i-1}), \forall i \in \boldsymbol{N} \tag{13}$$

where $Z_i(a_1, a_{i-1})$ is the overall cost of equipment *i*:

$$Z_i(a_1, a_{i-1}) = \begin{cases} K_i^m, & a_i = -1 \\ K_i^s(a), & a_i \ge 0 \end{cases} \tag{14}$$

Next, the above problem can be established as a game model with multi-users. $\Phi = (\mathrm{N}, \{A_i\}_{i \in N}, \{Z_i\}_{i \in N})$, where *N* is the set of mobile equipment participating in the game, A_i is the collection of participants' strategies, and the total cost function $Z_i(a_1, a_{i-1})$ is the cost function minimized by participant *i*. Φ is a game with multi-user attributes. Next, Nash equilibrium will be introduced in the game [27,28].

Definition 2. Strategy set $a^* = (a_1^*, \ldots, a_N^*)$ is a multi-user computing Nash equilibrium for offloading games. If there are no users in equilibrium a^*, they can unilaterally change their strategy to further reduce their overall cost, namely:

$$Z_i\left(a_i^*, a_{-i}^*\right) \le Z_i\left(a_i, a_{-i}^*\right), \forall a_i \in A_i, i \in N \tag{15}$$

According to the Nash equilibrium concept, for a game with multi-user attributes, if equipment i is both in Nash equilibrium a_i^* and chooses a cloud computing method (i.e., $a_i > 0$), an effective computation offloading strategy is necessary for equipment i [29].

4.2 Computation Offloading Algorithm

This algorithm enables the mobile equipment to realize mutually satisfactory computation offloading decisions before computing tasks are performed, achieving the Nash equilibrium after finite iterations by using the multi-user game model.

The clock signal of the wireless base station is used to synchronize the whole system, and the unloading decision is used to update the slot structure. Each time slot T consists of two steps:

(1) Receiving interference: if the i terminal device unloads to the edge server in the current time slot, it will send the pilot signal to the wireless base station S on its selected channel $a_i(t)$. Then, the wireless base station can measure the total received power $\rho_m(a_i(t))\sum_{n\in\blacksquare:a_n(t)} = mq_n g_{n,s}$). and respond with the power information to the i-th mobile user. Thus, the i-th end user mobile device i can obtain the received interference of all users on all channels:

$$\mu_n\left(m, a_{-i}(t)\right) = \begin{cases} \rho_m\left(a_i(t)\right) - q_n g_{n,s}, & a_i(t) = m \\ \rho_m a_i(t), & a_i(t) \ne m \end{cases} \tag{16}$$

The interference of the current channel $a_i(t)$ is equal to the power received by equipment i minus its own power, and the interference of other channels are equal to the power received by equipment i.

(2) Offloading updating: As the Nash equilibrium can be achieved by finite iterations, the mobile equipment is allowed to make decision updates for iteration. Based on interference $\{\rho_m(a_i(t), m \in M)\}$ measured on different channels, the set of equipment i with the best response update is firstly calculated as:

$$\Delta_i(t) \triangleq \left\{\tilde{a} : \tilde{a} = \arg\min_{a_i \in A_i} Z_i\left(a, a_{-i}(t)\right), Z_i\left(\tilde{a}, a_{-i}(t)\right) < Z_i\left(a_i(t), a_{-i}(t)\right)\right\} \tag{17}$$

If $\Delta_i(t) \ne \emptyset$ means that the i terminal user can improve the decision, the i terminal user will send a decision update request to the wireless base station. If $a_i(t) = 0$, it means that the i-th end user continues to maintain the original unloading decision, i.e., $a_i(t+1) = a_i(t)$. The wireless base station randomly selects one of all the terminal users who send the update request, and sends the update permission to the terminal. The terminal will update the decision to $a_i(t+1) \in \Delta_i(t)$ in the next time slot. All the terminals that have not received the update permission shall maintain the original decision.

5. Simulation Results and Analysis

5.1 Simulation Scenarios and Parameter Settings

This section uses MATLAB simulation software to verify the performance of the algorithm proposed in this section, and compares it with algorithms proposed in reference [19] and reference [20]. The

experiment is conducted on Win10 operating system with 8 G memory and Intel Core i7-8750H CPU @ 2.21 GHz. All simulation results are averaged over 500 independent simulations.

The system scenario is set in such a way that mobile users and MEC servers are randomly distributed in a circular area with a radius of 100 meters. The largescale channel fading model is $PL = d_{i,k}^{-\theta}$, where $d_{i,k}$ is the straight line distance between user i and MEC server k. The number of users is set to 30, and the number of MEC servers is set to 10. The Rayleigh fading is used as a small scale channel fading model. The constants k_0 and k_1 for the local CPU and the MEC server CPU are set to $k_0 = 1 \times 10^{-24}$ [30] and $k_1 = 1 \times 10^{-26}$ [31], respectively. In addition, other simulation parameters can be found in Table 1 unless otherwise specified.

5.2 Simulation Results

By identifying the Nash equilibrium state, the convergence of the proposed algorithm is analyzed, as shown in Fig. 2. As can be seen from this figure, with the increase in the number of iterations, the energy consumption shows a linear decreasing trend. At the number of 30 iterations, the proposed algorithm can achieve the Nash equilibrium. In other words, it can converge after a finite number of iterations.

Table 1. Simulation parameter settings

MEC system parameters	Value
Subcarrier width	$B_N = 15$ kHz
Background noise	$\delta^2 = -75$ dBm
Maximum transmission power	$P^m = 600$ mW
Input data size	$D_i = 1000$ bits
Task computational complexity	$X_i \in [1000,1200]$ cycles/bits
Task execution delay constraints	$\tau_i \in [9,10]$ ms
User CPU frequency	$f_{i,loc} \in [0.6,0.7]$ GHz
MEC server frequency	$f_{k,ser} \in [1.1,1.2]$ GHz

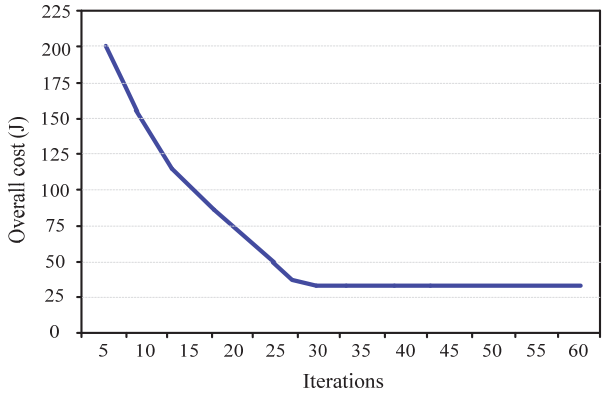

Fig. 2. Algorithm convergence analysis.

Fig. 3 evaluates the task execution cost, and compares our proposed algorithm with the algorithms in [19] and [20] under different conditions of noise power (background noises are -75, -80, -85, and -90 dBm, respectively); the relationship between task execution overheads and different amounts of task

input data is illustrated.

Fig. 3 shows that the task execution cost of our proposed strategy and the algorithms proposed in [19] and [20] all increase with the amount of task input data. As the increase in the amount of input data increases task execution delay and increases task execution energy consumption, this in turn will increase the task execution overheads. Furthermore, the task execution cost of the proposed algorithm is better than the task execution cost of algorithms proposed in [19] and [20]. The reason is that the algorithms proposed in [19] and [20] aim to optimize the task execution delay, which may lead to higher task execution energy consumption and increase the task execution overheads. The algorithm proposed in this paper models the distributed computation offloading decision between the mobile equipment as a multi-user computation offloading game, achieving a balance between the user diversity and the MEC server diversity.

Fig. 4 demonstrates the relationship between task execution overheads and the computing power of MEC servers, and shows the comparison among the algorithms proposed in [19] and [20]. It can be seen from this figure that the task execution cost of our proposed algorithm as well as of [19] and [20] all decrease with the increase of the computing power for MEC servers. The reason is that increasing the computing power of MEC servers can improve the task execution performance and reduce task execution overheads. By comparing the curves obtained by the three algorithms, it is evident that our proposed algorithm is superior to the algorithms proposed in [19] and [20].

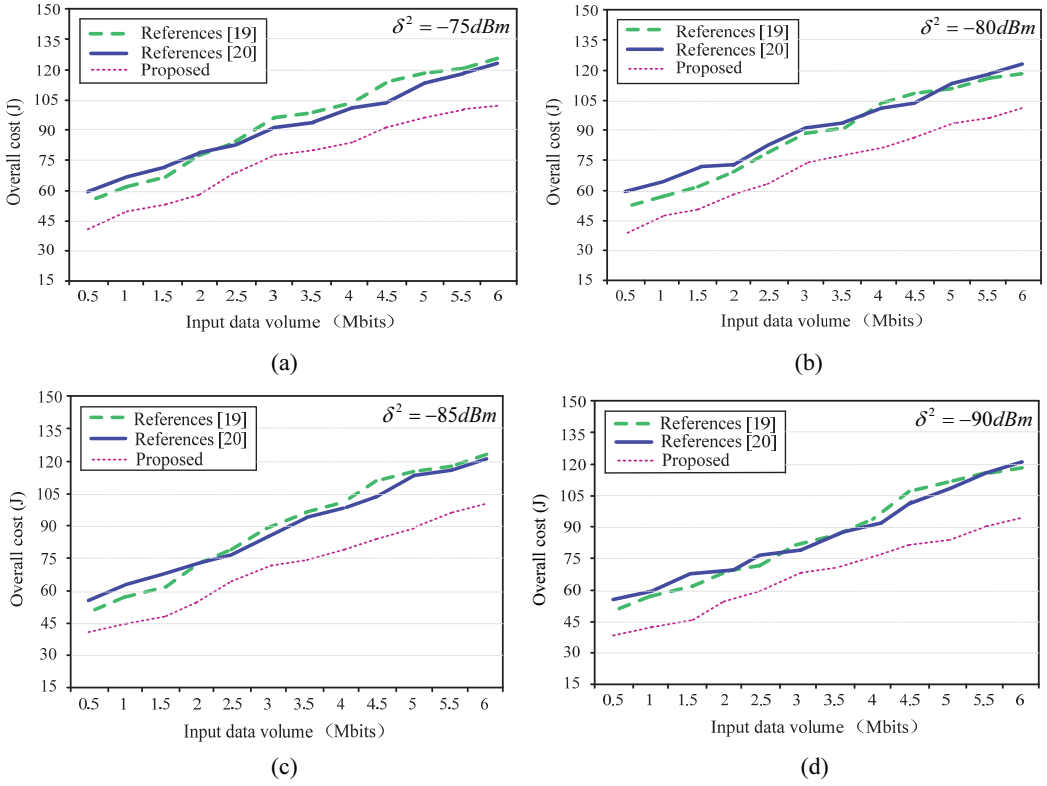

Fig. 3. The relationship between task execution overheads and task input data in different noises: (a) -75 dBm, (b) -80 dBm, (c) -85 dBm, and (d) -90 dBm.

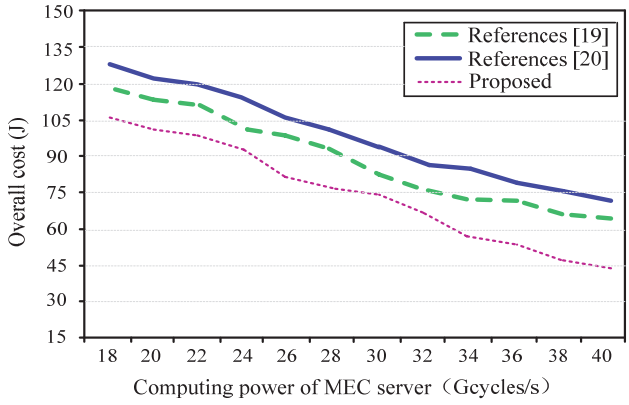

Fig. 4. The relationship between task execution overheads and the computing power of MEC servers.

Fig. 5 compares the relationship between the algorithm task execution cost and the number of requested users of our proposed algorithm and the two proposed in [19] and [20]. As can be seen from this figure, as the number of requested users increases, the task execution overheads increase. Compared with other algorithms, the proposed algorithm has the lowest task execution cost. The reason is that the proposed algorithm offloads mobile node's own computing tasks to edge servers. It comprehensively considers several factors, for example, energy consumption, delay, cost, resource usage, contribution or profit of different types of tasks, and reduces the overall overheads of computation offloading. This allows multi-users to efficiently perform the computational offloading under the game model. The performance gap of each algorithm increases as the number of requested users increases. The reason is that when the number of requested users increases, the resource competition at MEC servers will cause the system performance to decrease.

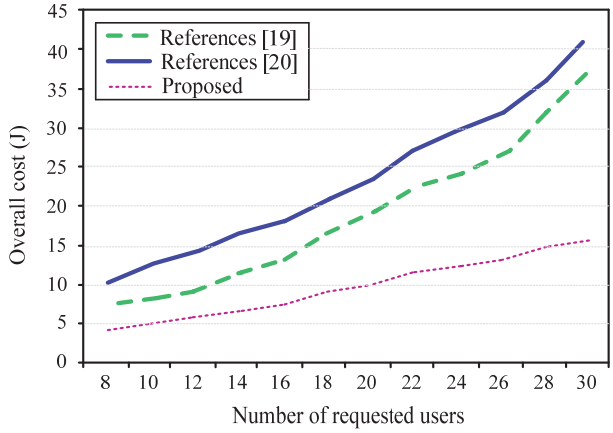

Fig. 5. The relationship between task execution cost and the number of requested users.

The algorithm designed in this paper first calculates the unloading weight, and then the distributed time slot is iterated to update the unloading decision. In summary, our proposed algorithm can effectively increase the number of useful decision-making users, and can achieve the balance by a limited number of iterations. Under the Nash equilibrium, the final decision strategy shows self-stability. The user has

no incentive to unilaterally change decisions, serving as a relatively stable final offloading decision plan for this period of time. At the same time, the proposed algorithm is superior to several other offloading algorithms in terms of the number of users and overall overheads for beneficial decision-making.

6. Conclusion

This paper discusses offloading strategies in a multi-user and multi-server scenario. In a resource-constrained system, we propose a multi-user and multi-server MEC task offloading algorithm based on cost optimization. The algorithm is verified by simulation experiments, and simulation results show that our proposed algorithm has a good offloading performance. In terms of the number of users and overall overheads for beneficial decision-making, the algorithm is superior to several other computation offloading algorithms. Notwithstanding, there are still a few problems that need further discussion and optimization. This paper assesses the task offloading weight by calculating the reward of tasks. The weight is mainly used to investigate the decision of mobile equipment to calculate locally or offload to edge servers. There is hardly any study of the impact of offloading weight in the case of performing service migration when the edge server has insufficient computing power. Therefore, it is suggested that future work can address the issue of service migration, and discuss the impact of offloading weight during service migration.

References

[1] Y. S. Jeong and J. H. Park, "Security, privacy, and efficiency of sustainable computing for future smart cities," *Journal of Information Processing Systems*, vol. 16, no. 1, pp. 1-5, 2020.

[2] W. Liu, L. Zhang, Z. Zhang, C. Gu, C. Wang, M. O'neill, and F. Lombardi, "XOR-based low-cost reconfigurable PUFs for IoT security," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 18, no. 3, pp. 1-21, 2019.

[3] V. Kumar, G. Sakya, and C. Shankar, "WSN and IoT based smart city model using the MQTT protocol," *Journal of Discrete Mathematical Sciences and Cryptography*, vol. 22, no. 8, pp. 1423-1434, 2019.

[4] W. Shi, X. Zhang, Y. Wang, and Q. Zhang, "Edge computing: state-of-the-art and future directions," *Journal of Computer Research and Development*, vol. 56, no. 1, pp. 69-89, 2019.

[5] E. Kim and S. Kim, "An efficient software defined data transmission scheme based on mobile edge computing for the massive IoT environment," *KSII Transactions on Internet & Information Systems*, vol. 12, no. 2, pp. 974-987, 2018.

[6] T. Quack, M. Bosinger, F. J. Heßeler, and D. Abel, "Infrastructure-based digital maps for connected autonomous vehicles," *at-Automatisierungstechnik*, vol. 66, no. 2, pp. 183-191, 2018.

[7] M. C. Filippou, D. Sabella, and V. Riccobene, "Flexible MEC service consumption through edge host zoning in 5G networks," in *Proceedings of 2019 IEEE Wireless Communications and Networking Conference Workshop (WCNCW)*, Marrakech, Morocco, 2019, pp. 1-6.

[8] K. Kanai, K. Imagane, and J. Katto, "Overview of multimedia mobile edge computing," *ITE Transactions on Media Technology and Applications*, vol. 6, no. 1, pp. 46-52, 2018.

[9] M. Li, F. R. Yu, P. Si, and Y. Zhang, "Green machine-to-machine communications with mobile edge computing and wireless network virtualization," *IEEE Communications Magazine*, vol. 56, no. 5, pp. 148-154, 2018.

[10] Q. Fan and N. Ansari, "Application aware workload allocation for edge computing-based IoT," *IEEE Internet of Things Journal*, vol. 5, no. 3, pp. 2146-2153, 2018.

[11] Y. Mao, J. Zhang, and K. B. Letaief, "Joint task offloading scheduling and transmit power allocation for mobile-edge computing systems," in *Proceedings of 2017 IEEE Wireless Communications and Networking Conference (WCNC)*, San Francisco, CA, 2017, pp. 1-6.

[12] C. Wang, F. R. Yu, C. Liang, Q. Chen, and L. Tang, "Joint computation offloading and interference management in wireless cellular networks with mobile edge computing," *IEEE Transactions on Vehicular Technology*, vol. 66, no. 8, pp. 7432-7445, 2017.

[13] A. De La Oliva, X. C. Perez, A. Azcorra, A. Di Giglio, F. Cavaliere, D. Tiegelbekkers, et al., "Xhaul: toward an integrated fronthaul/backhaul architecture in 5G networks," *IEEE Wireless Communications*, vol. 22, no. 5, pp. 32-40, 2015.

[14] J. Liu and Q. Zhang, "Offloading schemes in mobile edge computing for ultra-reliable low latency communications," *IEEE Access*, vol. 6, pp. 12825-12837, 2018.

[15] Q. D. Thinh, J. Tang, D. La Quang, and T. Q. Quek, "Adaptive computation scaling and task offloading in mobile edge computing," in *Proceedings of 2017 IEEE Wireless Communications and Networking Conference (WCNC)*, San Francisco, CA, 2017, pp. 1-6.

[16] G. Zhang and X. Liu, "Tasks split and offloading scheduling decision in mobile edge computing with limited resources," *Computer Applications and Software*, vol. 36, no. 10, pp. 268-278, 2019.

[17] S. S. Tanzil, O. N. Gharehshiran, and V. Krishnamurthy, "Femto-cloud formation: a coalitional game-theoretic approach," in *Proceedings of 2015 IEEE Global Communications Conference (GLOBECOM)*, San Diego, CA, 2015, pp. 1-6.

[18] 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. 8, pp. 3571-3584, 2017.

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

[20] L. Huang, X. Feng, C. Zhang, L. Qian, and Y. Wu, "Deep reinforcement learning-based joint task offloading and bandwidth allocation for multi-user mobile edge computing," *Digital Communications and Networks*, vol. 5, no. 1, pp. 10-17, 2019.

[21] Y. Wei, Z. Zhang, F. R. Yu, and Z. Han, "Joint user scheduling and content caching strategy for mobile edge networks using deep reinforcement learning," in *Proceedings of 2018 IEEE International Conference on Communications Workshops (ICC Workshops)*, Kansas City, MO, 2018, pp. 1-6.

[22] B. Bi and Y. J. Zhang, "Computation rate maximization for wireless powered mobile-edge computing with binary computation offloading," *IEEE Transactions on Wireless Communications*, vol. 17, no. 6, pp. 4177-4190, 2018.

[23] I. Dragan, "Egalitarian allocations and the inverse problem for the Shapley value," *American Journal of Operations Research*, vol. 8, no. 6, article no. 88284, 2018.

[24] W. Yang, J. Liu, and X. Liu, "Existence of fuzzy Zhou bargaining sets in TU fuzzy games," *International Journal of Fuzzy System Applications (IJFSA)*, vol. 7, no. 1, pp. 46-55, 2018.

[25] S. Sasikala, S. A. alias Balamurugan, and S. Geetha, "n efficient feature selection paradigm using PCA-CFS-Shapley values ensemble applied to small medical data sets," in *Proceedings of 2013 4th International Conference on Computing, Communications and Networking Technologies (ICCCNT)*, Tiruchengode, India, 2013, pp. 1-5.

[26] M. Gusev and S. Dustdar, "Going back to the roots: the evolution of edge computing, an iot perspective," *IEEE Internet Computing*, vol. 22, no. 2, pp. 5-15, 2018.

[27] J. Xu, Z. K. Yang, and W. Yuan, "Heterogeneous channel assignment of multi-radio multi-channel wireless networks: a game theoretic approach," *Journal of Chinese Computer Systems*, vol. 33, no. 5, pp. 1053-1056, 2012.

[28] Z. Huo, X. Li, S. Jin, and Z. Wang, "Nash equilibrium of an energy saving strategy with dual rate transmission in wireless regional area network," *Wireless Communications and Mobile Computing*, vol. 2017, article no. 9053862, 2017. https://doi.org/10.1155/2017/9053862

[29] Y. Yang, Y. Li, W. Zhang, F. Qin, P. Zhu, and C. X. Wang, "Generative-adversarial-network-based wireless channel modeling: challenges and opportunities," *IEEE Communications Magazine*, vol. 57, no. 3, pp. 22-27, 2019.

[30] K. De Vogeleer, G. Memmi, P. Jouvelot, and F. Coelho, "The energy/frequency convexity rule: modeling and experimental validation on mobile devices," in *Parallel Processing and Applied Mathematics*. Heidelberg, Germany: Springer, 2013, pp. 793-803.

[31] F. Wang, J. Xu, X. Wang, and S. Cui, "Joint offloading and computing optimization in wireless powered mobile-edge computing systems," *IEEE Transactions on Wireless Communications*, vol. 17, no. 3, pp. 1784-1797, 2018.

Yanfei He **https://orcid.org/0000-0003-3789-7926**

She is a Master of Geographic Information System and a lecturer. She graduated from SiChuan Normal University in 2009. She is working lecturer in Zhejiang Yuying College of Vocational Technology now. Her research interests include Internet of Things and higher vocational education of computer.

Zhenhua Tang **https://orcid.org/0000-0001-6405-695X**

He is a Master of Computer Science and a senior engineer. He graduated from Hangzhou Dianzi University in 2010. He is working in Zhejiang Radio and Television Group. His research interests include media convergence technology and Internet of Things.