Joint Optimization Offloading Strategy of Execution Time and Energy Consumption of Mobile Edge Computing

Qingzhu Wang and Xiaoyun Cui
School of Computer Science, Northeast Electric Power University, China

Abstract: *As mobile devices become more and more powerful, applications generate a large number of computing tasks, and mobile devices themselves cannot meet the needs of users. This article proposes a computation offloading model in which execution units including mobile devices, edge server, and cloud server. Previous studies on joint optimization only considered tasks execution time and the energy consumption of mobile devices, and ignored the energy consumption of edge and cloud server. However, edge server and cloud server energy consumption have a significant impact on the final offloading decision. This paper comprehensively considers execution time and energy consumption of three execution units, and formulates task offloading decision as a single-objective optimization problem. Genetic algorithm with elitism preservation and random strategy is adopted to obtain optimal solution of the problem. At last, simulation experiments show that the proposed computation offloading model has lower fitness value compared with other computation offloading models.*

Keyword: *Energy consumption, execution time, mobile edge computing, offloading strategy.*

Received July 15, 2020; accepted March 10, 2021
https://doi.org/10.34028/iajit/18/5/11

1. Introduction

In the era of Internet of things [1, 4, 13], smart mobile terminal devices have become an essential part of people's daily life. However, they have their limitations, such as limited battery capacity, low computing and storage capacity. Cloud computing [3, 7, 13, 16] with scalable computing and storage resource was adopted to solve these problems. But the delay of cloud computing is too large to meet the needs of the Internet of things, and then mobile edge computing [2, 11, 12, 18] is utilized. Since mobile edge computing deploys servers closer to users, it can effectively reduce latency while extending smart device processing capacity.

Zhang *et al*. [23] presented an iterative search algorithm combining interior penalty function with the difference of two convex functions/sets programming to obtain the optimal solution of mixed integer nonlinear problem for computation offloading and resource allocation. But the execution units only included mobile devices and edge server. Moreover, in the multiple mobile edge computing networks, final solutions obtained by iterative search algorithm is suboptimal rather than optimal. Liu *et al*. [8, 9] put forward a tensor-based representation model to comprehensively reflect the complex relationship of multiple influencing factors and cope with their heterogeneity. But how to improve the optimization efficiency and reduce the search space through tensor-based dimensionality reduction approach are complex. Hoang *et al*. [6] studied the mobile edge offloading scenario consisting of one mobile device with independent tasks and various remote edge devices providing computing resource. Then the user's device can offload the tasks to available access points for edge computing. They jointly optimized task allocation decision and data compression ratio to minimize the execution time of total tasks and energy consumption of mobile device. Nevertheless, in their system model, there is only one terminal device with tasks, which is almost impossible to happen in real world. Xu *et al*. [20] analyzed execution time of tasks and energy consumption of the mobile devices, then formulated them as a multi-objective optimization problem. And they adopted non-dominated sorting genetic algorithm III to address the multi-objective optimization problem. However, they only focused on the execution of tasks and the energy consumption of mobile devices, and ignored the energy consumption of edge servers and cloud servers.

In fact, the two-tier model already has related applications, like 5G, intelligent video surveillance system. In this paper, we collaboratively study computation offloading strategy under the scenario of mobile devices, edge server and cloud server. Compared with the two-tier system model of "cloud server and mobile device", the three-tier system model we proposed has lower latency and smaller jitter, and is more suitable for latency-sensitive applications, such as automatic driving and intelligent medical treatment. And compared with the two-tier system model of "edge server and mobile device", our system model has

more powerful computing power, and has better effect for the application with large amount of computation. How to reduce execution time and energy consumption is our research object when processing tasks. The major works of this study are summarized as follows:

1. We propose a three-layers system model for mobile edge computing, namely, mobile device layer, edge server layer and cloud server layer. Computation tasks generated by mobile devices can be executed locally, or offloaded to edge server or cloud server according to offloading strategy. What's more, the time and energy consumption models are represented respectively.
2. We formulate execution time and energy consumption as a single-objective function. Therein, execution time is the total time of all tasks, and energy consumption is execution and transmission energy including mobile devices, edge and cloud server.
3. Genetic algorithm with random strategy and elitism preservation is used to resolve this single-objective optimization problem, and experiment result verifies its effectiveness.

The remainder of this paper is organized as follows. System model and the problem formulation are proposed in sections 2 and 3, a computation offloading method of mobile edge computing is elaborated. Section 4 evaluates the proposed method. And section 5 gives the conclusion and the future work.

2. System Model and Problem Formulation

2.1. System Model

Figure 1 illustrates a system framework for mobile edge computing. In this framework, we consider a scenario where an edge server domain covers M mobile devices which are connected to the edge server by Local Area Network (LAN) [8] and a cloud server domain covers Q edge servers which are connected to the cloud server by Wide Area Network (WAN) [9]. In this paper, the computing tasks are available to be executed by the mobile devices, edge server or cloud server according to computation offloading strategy. And it is up to the mobile device to decide whether to offload tasks and how many computing tasks to offload. In addition, tasks are only offloaded to the edge server which covers the domain of devices generating these tasks, leaving aside the offloading of tasks between edge servers.

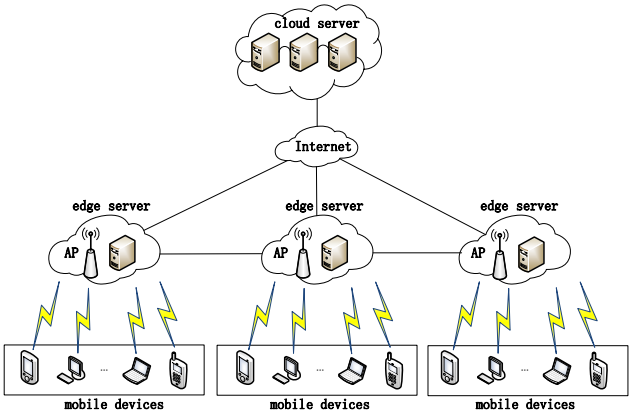


Figure 1. System model of mobile edge computing.

2.2. Task Model

Suppose a mobile device has N calculation tasks to be performed. The tasks, which are independent of each other, generated by the same mobile device can be formalized as a set. Let t_m^n ($m = \{1, 2, \ldots, M\}$, $n = \{1,2, \ldots, N\}$) be the nth task generated by the mth mobile device, where t_m^n is a triple that can be expressed as $\{ f_m^n, u_m^n, d_m^n, l_m^n \}$, where f_m^n denotes the number of CPU cycles consumed to calculate the task, u_m^n is the amount of data size of task t_m^n when the task offloads to edge server or cloud server, d_m^n represents the amount of return data size of the calculation result, and l_m^n signifies the maximum deadline of task t_m^n.

A task can be executed locally, a core of edge server or a core of cloud server according to the offloading decision. For simplicity, every mobile device has only one core that has the same processing capacity. The core of mobile devices is equipped with a Dynamic Voltage and Frequency Scaling (DVFS) [10, 17] capability such that each core can operate at L different frequency levels with the corresponding L supply voltage levels. As mentioned above, the maximum operating frequency of the core is f_{max}, and there are L frequency scaling factors, $a_{k,1}<\ldots<a_{k,l}<\ldots<a_{K,L}$. Therefore, actual operating frequency of mobile device can be denoted as $f= a_{k,l} \cdot f_{max}$. The power of mobile device can be expressed as $P_{lc}= \alpha_m \cdot f^{\gamma}$, in this paper, the value of γ is equal to 2 [10]. In addition, when mobile device is in the task transmission stage, the power consumption of the mobile device can be expressed as P_{lt}. And since the power consumption of the terminal in idle state is inevitable, it is ignored in this paper.

If task t_m^n is offloaded to edge server on the basis of offloading strategy, there are three phases in sequence associated with the execution of task t_m^n:

1. Wireless transmission phase.
2. Edge computing phase.

3. Wireless transmission return phase.

In the wireless transmission phase, the input data of task t_m^n is sent to edge server through wireless channel. In the edge computing phase, task is executed in edge server. In some applications, output data sizes are near to input ones, such as infotainment app, heavy compare app and so on. It is unreasonable to ignore the amount of data returned. In wireless transmission return phase, the mobile device receives the output data of task from the edge server through the wireless channel. And edge server transmits the output data of task back to the mobile device as long as it finishes processing task t_m^n. We use P_{et} to denote the power consumption of edge server in wireless transmission phase for sending or receiving data between the mobile and edge server, and P_{ec} to denote the power consumption of edge server in edge computing phase.

If task t_m^n is offloaded to cloud server according to offloading decision, there are five phases in sequence associated with the execution of task t_m^n:

1. Wireless transmission phase.
2. Wired transmission phase.
3. Cloud computing phase.
4. Wired transmission return phase.
5. Wireless transmission return phase.

Wireless transmission and return phase are same as the process mentioned above. The wired transmission and wired return phases are respectively the process of transferring task data from edge server to cloud server and return data from cloud service to edge server through the wired channels. We use P_{ed} to represent the power consumption of edge server in wired transmission and return phase. Since the cloud server may handle multiple tasks delivered by multiple edge servers at the same time, it is not accurate to calculate the energy consumption of cloud server according to the transmission or running power of the server. In this paper, we use energy consumption W_{cc} to denote the energy consumption of cloud server for executing 1 Central Processing Unit (CPU) cycle task, and W_{ct} to represent the energy consumption of cloud server for transmitting 1bit data.

2.3. Execution Time and Energy Consumption Model

Next, we will formulate execution time and energy consumption for tasks to be performed on mobile devices, edge server, and cloud server, respectively [18, 23].

2.3.1. Execution Time Model

In this paper, $x_m^n = 0$ if task is assigned to be executed locally, and if task is offloaded to edge or cloud server, $x_m^n = 1$.Then the execution time of task t_m^n executed on the mobile device is

$$T_l\left(t_m^n\right) = \frac{f_m^n}{f_{max}} \tag{1}$$

When task t_m^n is assigned edge server for execution, task execution time $T_e\left(t_m^n\right)$ includes the time of task calculation on edge server $T_{ec}\left(t_m^n\right)$ and the time of data transmission through wireless channel $T_{et}\left(t_m^n\right)$, and $T_{et}\left(t_m^n\right)$ involves data transmission time and data return time. Then $T_{et}\left(t_m^n\right)$, $T_{ec}\left(t_m^n\right)$ and $T_e\left(t_m^n\right)$ can be respectively expressed as

$$T_{et}\left(t_m^n\right) = \frac{u_m^n}{c_l} + \frac{d_m^n}{c_l} \tag{2}$$

$$T_{ec}\left(t_m^n\right) = \frac{f_m^n}{f_{mec}} \tag{3}$$

$$T_e\left(t_m^n\right) = T_{et}\left(t_m^n\right) + T_{ec}\left(t_m^n\right) \tag{4}$$

Where c_l is the bandwidth of wireless channel. For simplicity, it is same to upload or download. f_{mec} is used to represent processing capacity of edge server, which is measured by CPU cycles.

If task t_m^n is executed on cloud server according to offloading decision, task execution time includes five parts: wireless uploading time, wired uploading time, task execution time, wired downloading time and wireless downloading time. Let c_d be the bandwidth of wired channel, then task execution time $T_c\left(t_m^n\right)$ can be divided into three parts: the time of data transmission through wireless channel $T_{ct}\left(t_m^n\right)$, the time of data transmission through wired channel $T_{cd}\left(t_m^n\right)$, and the time of task calculation on cloud server $T_{cc}\left(t_m^n\right)$, they are severally represented by

$$T_{ct}\left(t_m^n\right) = \frac{u_m^n}{c_l} + \frac{d_m^n}{c_l} \tag{5}$$

$$T_{cd}\left(t_m^n\right) = \frac{u_m^n}{c_d} + \frac{d_m^n}{c_d} \tag{6}$$

$$T_{cc}\left(t_m^n\right) = \frac{f_m^n}{f_{cc}} \tag{7}$$

$$T_c\left(t_m^n\right) = T_{ct}\left(t_m^n\right) + T_{cd}\left(t_m^n\right) + T_{cc}\left(t_m^n\right) \tag{8}$$

For the computing task t_m^n, adopting computation offloading strategy x_m^n, y_m^n, the execution time TL_m^n of task t_m^n can be denoted by

$$TL_m^n = \begin{cases} T_l\left(t_m^n\right), x_m^n = 0, y_m^n = 0 \\ T_e\left(t_m^n\right), x_m^n = 1, y_m^n = 0 \\ T_c\left(t_m^n\right), x_m^n = 1, y_m^n = 1 \end{cases} \tag{9}$$

Where $x_m^n = 0, y_m^n = 0$ represents t_m^n is executed on mobile device, $x_m^n = 1, y_m^n = 0$ indicates t_m^n is

offloaded to edge server, and $x_m^n = 1, y_m^n = 1$ denotes t_m^n is offloaded to cloud server.

Based on the offloading decision, T_L is the execution time of all tasks assigned to be executed locally, T_E denotes the time it takes to complete all tasks offloaded to edge server, and T_C is the time to execute all tasks offloaded to cloud server.

$$T_L = \sum_{m=1}^{M}\sum_{n=1}^{N}\left(1-x_m^n\right)\cdot\left(1-y_m^n\right)\cdot T_l\left(t_m^n\right) \tag{10}$$

$$T_E = \sum_{m=1}^{M}\sum_{n=1}^{N}x_m^n\cdot\left(1-y_m^n\right)\cdot T_e\left(t_m^n\right) \tag{11}$$

$$T_C = \sum_{m=1}^{M}\sum_{n=1}^{N}x_m^n\cdot y_m^n\cdot T_c\left(t_m^n\right) \tag{12}$$

To sum up, the total execution time T_{total} of all tasks can be expressed as

$$T_{total} = T_L + T_E + T_C \tag{13}$$

2.3.2. Energy Consumption Model

Reducing energy consumption of the whole system is another goal of our study.

For task t_m^n, energy consumption of mobile device $E_l\left(t_m^n\right)$ can be denoted by

$$\begin{aligned}E_l\left(t_m^n\right) &= \left(1-x_m^n\right)\cdot P_{lc}\cdot T_l\left(t_m^n\right) + x_m^n\cdot\left(1-y_m^n\right)\\ &\cdot P_{lt}\cdot T_{et}\left(t_m^n\right) + x_m^n\cdot y_m^n\cdot P_{lt}\cdot T_{ct}\left(t_m^n\right)\end{aligned} \tag{14}$$

For task t_m^n, energy consumption of edge server $E_e\left(t_m^n\right)$, can be calculated by

$$\begin{aligned}E_e\left(t_m^n\right) &= x_m^n\cdot\left(1-y_m^n\right)\cdot P_{et}\cdot T_{et}\left(t_m^n\right) + x_m^n\\ &\cdot\left(1-y_m^n\right)\cdot P_{ec}\cdot T_{ec}\left(t_m^n\right) + x_m^n\cdot y_m^n\cdot P_{ed}\cdot T_{cd}\left(t_m^n\right)\end{aligned} \tag{15}$$

And for task t_m^n, energy consumption of cloud server $E_c\left(t_m^n\right)$, can be computed by

$$\begin{aligned}E_c\left(t_m^n\right) &= x_m^n\cdot y_m^n\cdot W_{cc}\cdot f_m^n\\ &+ x_m^n\cdot y_m^n\cdot W_{ct}\cdot\left(u_m^n + d_m^n\right)\end{aligned} \tag{16}$$

Therefore, the total energy of all tasks can be formulated as

$$E_{total} = \sum_{m=1}^{M}\sum_{n=1}^{N}\left(E_l\left(t_m^n\right) + E_e\left(t_m^n\right) + E_c\left(t_m^n\right)\right) \tag{17}$$

2.4. Problem Formulation

In this paper, we intend to shorten the execution time, given in (13) and save the energy consumption of the whole system, presented in (17). The formalized problem can be defined as

$$\min\quad F = \alpha\cdot T_{total} + \left(1-\alpha\right)\cdot E_{total} \tag{18}$$

s. t.

$$\begin{aligned}&\forall max\left\{T_l\left(t_m^n\right), T_e\left(t_m^n\right), T_c\left(t_m^n\right)\right\} \le l_m^n\\ &m = \{1,2,\cdots,M\}, n = \{1,2,\cdots,N\}\end{aligned} \tag{19}$$

$$\alpha \in [0,1] \tag{20}$$

$$\left(x_m^n, y_m^n\right) \in \{(0,0),(1,0),(1,1)\} \tag{21}$$

It is our optimization objective to achieve overall equilibrium state of calculation time and energy consumption of various devices. By formula (18), the multi-objective optimization problem can be transformed into a single-objective optimization problem. And the coefficient α can be adjusted according to the different needs of users. When α equals to 0, it is mainly to obtain the minimum value of energy consumption of the whole system, while α equals to 1, it is to find the minimum value of the overall execution time.

3. Computation Offloading Method

In this section, computation offloading decision of $M \cdot N$ tasks generated by M mobile devices is defined as a single-objective optimization problem of shortening the executing time of tasks and saving the energy consumption of all devices, including mobile devices, edge server, and cloud server.

Firstly, we randomly encode the computation offloading strategy in the form of "00", "10", "11". Then the fitness function of optimization objective as well as constraints is formulated. What's more, genetic algorithm with elitism preservation and random strategy is adopted to obtain optimal solution of the problem.

In order to improve the shortcoming that genetic algorithm [22] is prone to fall into the local optimal solution, generation of offspring in this paper includes three parts: parent generation selected by roulette, elite individual inherited from the parent generation and a group of randomly generated solutions. Besides, we randomly choose single point, midpoint, two-point or multi-point crossover at the stage of genetic crossover. And in the phase of mutation, the strategy of random variation number is used to increase the ability of getting globally optimal solution of genetic algorithm. The purpose of this paper is to get the minimum fitness value, so roulette wheel selection operator changes. The less the fitness value is, the more likely to be chosen. Cumulative fitness is expressed as *sum*, the total number of chromosomes is denoted as *I*, and F_i is used to represent the fitness value of ith chromosome. Then the probability of ith chromosome being selected can be expressed as

$$PR_i = \frac{sum - F_i}{sum\cdot(I-1)}, i = (1,2,\cdots,I) \tag{22}$$

The specific process is as follow:

Algorithm1: Genetic algorithm with elitism preservation and random strategy

Input:
population: I,
number of iterations: T,

crossover probability: c,
mutation probability: m,
computational task set:
$TS = \{t_1^1, t_1^2, \cdots, t_1^N, t_2^1, t_2^2, \cdots, t_2^N, \cdots, t_M^1, t_M^2, \cdots, t_M^N\}$
operating frequency: f_{max}, f_{mec}, f_{cc},
power consumption of the device: P_{lc}, P_{lt}, P_{et}, P_{ec}, P_{ed}, P_{cc}, P_{ct}
channel bandwidth: c_l, c_d
parameter coefficient: α
generate an initial population Pop by random method
do {
calculate function value of each chromosome in the population by (18), and it satisfies the constraints (19-21)
initialize the null population newPop
save the fitness value of minimum fitness and the offloading decision at this fitness
do {
two individuals (p1, p2) are selected from the population by section operator
if (random (0, 1) < c) {
randomly generate integers r from 0 to 3
if (r = 0) {
single point crossover
} else if (r = 1) {
midpoint crossover
} else if (r = 2) {
two-point crossover
} else (r = 3) {
multi-point crossover
}
}
if (random (0, 1) < m) {
randomly generate integers s1 from 1 to the length of chromosome
randomly generate integers s2 from 1 to the length of chromosome
for p1 randomly select s1 positions for mutation operation
for p2 randomly select s1 positions for mutation operation
}
add these two new individuals to newPop
} until (I offspring are created)
replace Pop with newPop
} until (meet the termination conditions)
return optimal offloading decision, execution time, energy consumption and optimal fitness.

4. Experimental Evaluation

In this part, we evaluate the performance of the proposed computation offloading method, and compare with other methods by comprehensive simulations and experiments.

In the simulation experiment, execution time, energy consumption, and the fitness value of different task scheduling schemes are compared. And Some experimental parameters were set according to references [15, 18].

The parameter setting is shown in Tables 1 and 2.

Table 1. Simulation parameters.

Parameter	Value	Parameter	Value
f_m^n	[1000,15000]M	P_{lt}	0.2V
u_m^n	[50,700] Kbit	P_{ec}	1V
d_m^n	[20,700] Kbit	P_{et}	0.4V
f_{max}	0.4GHZ	P_{ed}	0.5V
f_{mec}	1.5GHZ	W_{cc}	10^{-10}W/cycle
f_{cc}	4GHZ	W_{ct}	$5{\cdot}10^{-7}$W/bit
α_m	3.125	c_l	100Kbps
P_{lc}	0.5V	c_d	200Kbps

Table 2. Parameters of the genetic algorithm.

parameter	value	parameter	value
c	0.8	T	100
m	0.3	α	0.5

We compare our proposed offloading method (The proposed) with local execution (Local), local and edge collaborative offloading decision [5] (Local-Edge), local and cloud collaborative offloading strategy [19] (Local-Cloud). There are ten mobile devices which in the same edge server domain. Every mobile device generates one to ten computation tasks.

In Figure 2, execution time increases with increase of the number of tasks. When the task is executed locally, the execution time is significantly higher than that of other methods. That is because mobile devices have little computing capacity and cannot complete the computing tasks in a timely and efficient manner when a large number of tasks are performed locally. Furthermore, Local-Cloud takes longer to complete tasks than Local-Edge because the cloud server is farther away from the user than the edge server. And the method we proposed takes the least time.

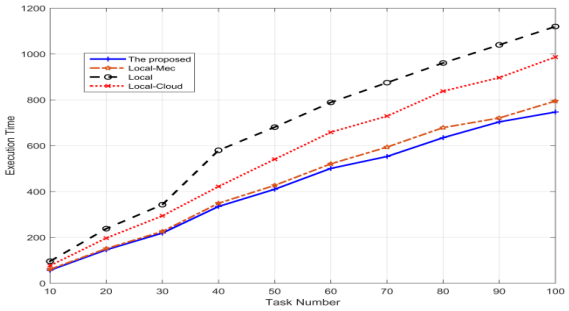


Figure 2. Execution time of different methods.

And we can see from Figure 3, energy consumption increases with the increase of the number of tasks, too. When tasks are executed locally, the total energy consumption is that of mobile devices. Tasks performed locally cost the most energy, and excessive energy consumption of mobile devices will affect user experience and shorten battery life of terminal devices. Due that the calculation and transmission energy consumption are calculated separately according to the tasks, and the energy consumption of the equipment itself is not included, Local-Cloud costs less energy than Local-Edge.

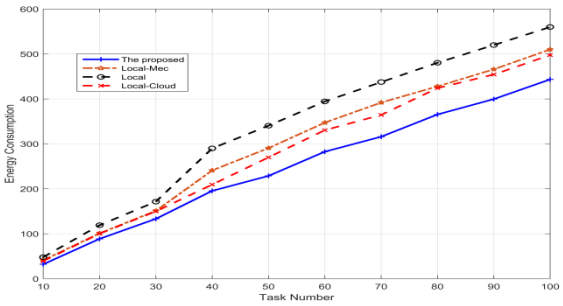


Figure 3. Energy consumption of different methods.

According to the results of previous experiments, both execution time and energy consumption increase with the number of tasks, then as the number of tasks increases, so does the total fitness value. The same result can be seen in Figure 4. And the model of Local-Mec works better than Local-Cloud, our proposed model is best in these offloading methods.

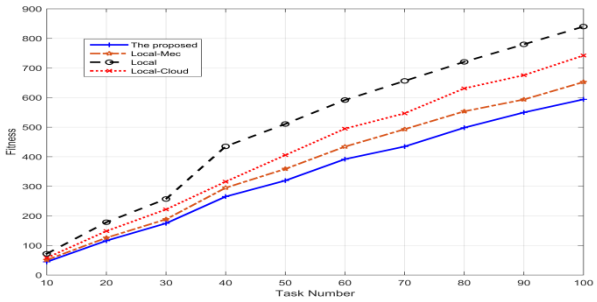


Figure 4. Fitness of different methods.

As shown in Figure 5, the fitness value of our proposed computation offloading model is lower than other methods When α is taking different values. It fully proves that our method is superior to other methods. Users can modify the value of coefficient α according to different requirements.

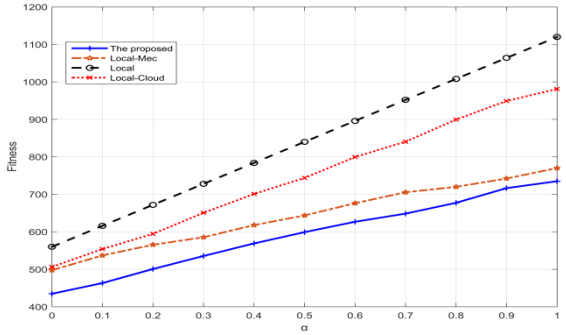


Figure 5. Fitness of different value of α.

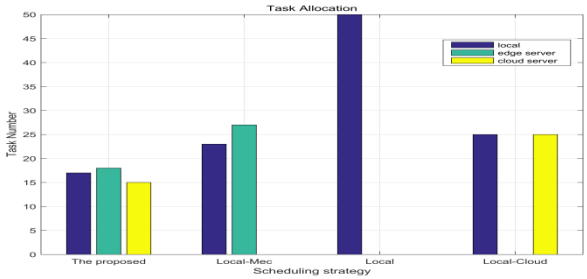
Table 3 shows the fitness values of different offloading strategies when the number of tasks is 20, 40, 60, 80, and 100 respectively, and α=0.5. From Table 3, we can see that the offloading strategy proposed in this paper can obtain a smaller fitness value. Table 4 shows the fitness values of different offloading strategies when the number of tasks is 100 and α is 0.1, 0.3, 0.5, 0.7, and 0.9 respectively. From Table 4, we can see that the offloading strategy proposed in this paper is better than other offloading strategies.

Table 3. Fitness of different computing offloading model.

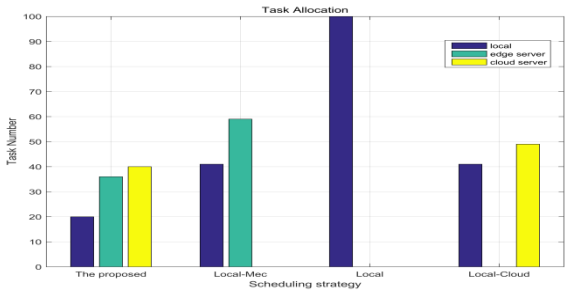
Task number / Algorithm	20	40	60	80	100
The proposed	116.11	265.25	391.83	498.14	593.98
Local-Mec	126.23	295.10	434.15	553.45	652.55
Local	178.50	435.42	591.75	720.75	840.00
Local-Cloud	148.58	315.73	494.78	631.03	742.34

Table 4. Fitness of different computing offloading model.

α / Algorithm	0.1	0.3	0.5	0.7	0.9
The proposed	463.26	535.47	599.09	648.5	716.56
Local-Mec	537.01	585.45	643.65	705.49	742.28
Local	616.15	728.18	840.34	952.82	1064.04
Local-Cloud	554.12	650.8	743.9	840.43	949.11



a) The task distribution of the 50 tasks.



b) The task distribution of the 100 tasks.

Figure 6. The distribution of task allocation.

When α=0.5, the number of tasks generated by per mobile device is 5, the task distribution in different allocation scheme is shown in Figure 6-a). Except that the number of tasks is 10, task assignments are illustrated in Figure 6-b). As the number of tasks generated by mobile devices increases, the proportion of tasks assigned to terminal devices decreases. This is because mobile devices have limited processing capacity and cannot meet the maximum delay requirement when the number of tasks increases. As a result, more tasks are assigned to edge or cloud server for execution.

In order to verify the superiority of our proposed improved genetic algorithm, we will compare the improved algorithm with binary Particle Swarm Optimization Algorithm (BPSO) [14] and ant Colony Optimization (ACO) [21] on the basis of the three-layer computation offloading model proposed in this paper. α is equal to 0.5.

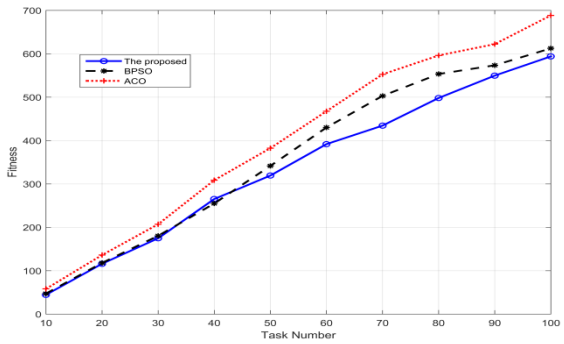


Figure 7. Comparison of different algorithms.

As can be seen from the Figure 7, with the increasing number of tasks, the fitness value is also increasing. And the algorithm we proposed is better than other comparison algorithms in general under the same conditions. When the number of tasks is relatively small, the effect is not very obvious, and with the increasing number of tasks, the proposed algorithm is more efficient.

Table 5. Fitness value of the different algorithms.

Time / Algorithm	1	2	3	4	5
GA	595.23	587.32	594.63	590.46	589.22
BPSO	622.84	619.46	625.81	618.89	620.81
ACO	662.35	654.37	667.13	659.37	664.25

For the problem that the algorithm is sensitive to the initial value, we repeated the experiment for many times. In the process of experiment, the initial value is generated randomly, and the probability of "00", "10" and "11" is the same. When the number of tasks is 100, α=0.5, the fitness values of different algorithms are randomly selected for 5 times, as shown in Table 5. Although the initial values of algorithms are different, they have little influence on the experiment results in the application of task scheduling. Besides, our strategy for selecting initial values is to randomly generate within the range of values.

5. Conclusions

This paper proposes a system model, which comprehensively considers calculation, transmission time as well as energy consumption of mobile devices, edge and cloud server, and jointly carry out tasks offloading decision. In addition, genetic algorithm with elitism preservation and random strategy is used to find optimal solution. Experiment results show that the proposed method is stable as the value of α changes. And it can provide a computational offloading scheme with optimal fitness. The algorithm used in this paper is genetic algorithm, the complexity of this algorithm is $T(n)=O(N\cdot T)\cdot O(n^2)$, Where N is the population size and T is the maximum number of iterations. In practice, the two-tier model [2, 3, 13, 18] already has related applications, like 5G, intelligent video surveillance system. The hardware conditions of the scheme can be met, but the software needs further research and development, and the algorithm used is not very complex, so it is not difficult to use in practice. In this paper, the tasks have no dependencies with each other. However, in the actual calculation, a large proportion of tasks are related. Next, we will consider the time and energy balance of dependent tasks.

Reference

[1] Abou-Tair D., Büchsenstein S., and Khalifeh A., "A Fog Computing-based Framework for Privacy Preserving IoT Environments," *The International Arab Journal of Information Technology*, vol. 17, no. 3, pp. 306-315, 2020.

[2] Ahmed A. and Ahmed E., "A Survey on Mobile Edge Computing," *in Proceeding of 10th IEEE International Conference on Intelligent Systems and Control*, Coimbatore, pp. 1-8, 2016.

[3] Arunarani A., Manjula D., and Sugumaran V., "Task Scheduling Techniques in Cloud Computing: A Literature Survey," *Future Generation Computer Systems*, vol. 91, pp. 407-415, 2019.

[4] Banijamali A., Pakanen O., Kuvaja P., and Oivo M., "Software Architectures of the Convergence of Cloud Computing and the Internet of Things: A Systematic Literature Review," *Information and Software Technology*, vol. 122, pp. 1-24, 2020.

[5] Cui L., Xu C., Yang S., Huang J, and Lu N., "Joint Optimization of Energy Consumption and Latency in Mobile Edge Computing for Internet of Things," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4791-4803, 2019.

[6] Hoang M., Dinh T., and Hoang H., "Joint Optimization of Execution Latency and Energy Consumption for Mobile Edge Computing with Data Compression and Task Allocation," *in Proceeding of International Symposium on Electrical and Electronics Engineering*, Ho Chi Minh, pp. 113-118, 2019.

[7] Kumar M., Sharma SC., Goel A., and Singh SP., "A Comprehensive Survey for Scheduling Techniques in Cloud Computing," *Journal of Network and Computer Applications*, vol. 143, pp. 1-33, 2019.

[8] Liu H., Pu J., Yang L., Lin M., Yin D., Guo Y., and Chen X., "A Holistic Optimization Framework for Mobile Cloud Task Scheduling," *IEEE Transactions on Sustainable Computing*, vol. 4, no. 2, pp. 217-230, 2019.

[9] Liu H., Yang L., Lin M., Yin D., and Guo Y., "A Tensor-Based Holistic Edge Computing Optimization Framework for Internet of Things," *IEEE Network*, vol. 32, no. 1, pp. 88-95, 2018.

[10] Lin X., Wang Y., Xie Q., and Pedram M., "Task Scheduling with Dynamic Voltage and

Frequency Scaling for Energy Minimization in the Mobile Cloud Computing Environment," *IEEE Transactions on Services Computing*, vol. 8, no. 2, pp. 175-186, 2015.

[11] Mach P. and Becvar Z., "Mobile Edge Computing: A Survey on Architecture and Computation Offloading," *IEEE Communications Surveys and Tutorials*, vol. 19, no. 3, pp. 1628-1656, 2017.

[12] Mao Y., You C., Zhang J., Huang K., and Letaief K., "A Survey on Mobile Edge Computing: The Communication Perspective," *IEEE Communications Surveys and Tutorials*, vol. 19, no. 4, pp. 2322-2358, 2017.

[13] Nord J., Koohang A., and Paliszkiewicz J., "The Internet of Things: Review and Theoretical Framework," *Expert Systems with Applications*, vol. 133, pp. 97-108, 2019.

[14] Pegado R., Naupari Z., Molina Y., and Castillo C., "Radial Distribution Network Reconfiguration for Power Losses Reduction Based on Improved Selective BPSO," *Electric Power Systems Research*, vol. 169, pp. 206-213, 2019.

[15] Peng H., Wen W., Tseng M., and Li L., "Joint Optimization Method for Task Scheduling Time and Energy Consumption in Mobile Cloud Computing Environment," *Applied Soft Computing*, vol. 80, pp. 534-545, 2019.

[16] Sharma Y., Javadi B., Si W., and Sun D., "Reliability and Energy Efficiency in Cloud Computing Systems: Survey and Taxonomy," *Journal of Network and Computer Applications*, vol. 74, pp. 66-85, 2016.

[17] Shirvani M., Rahmani A., and Sahafi A., "A Survey Study on Virtual Machine Migration and Server Consolidation Techniques in DVFS-Enabled Cloud Datacenter: Taxonomy And Challenges," *Journal of King Saud University-Computer and Information Sciences*, vol. 32, no. 3, pp. 267-286, 2020.

[18] Tong Z., Deng X., Ye F., Basodi S., and Pan Y., "Adaptive Computation Offloading and Resource Allocation Strategy in a Mobile Edge Computing Environment," *Information Sciences*, vol. 537, pp. 116-131, 2020.

[19] Vankadara S. and Dasari N., "Energy-aware Dynamic Task Offloading and Collective Task Execution in Mobile Cloud Computing," *Wiley*, pp. 1-14, 2019.

[20] Xu X., Liu Q., Luo Y., Peng K., Zhang X., Meng S., and Qi L., "A Computation Offloading Method over Big Data for IoT-enabled Cloud-edge Computing," *Future Generation Computer Systems*, vol. 95, pp. 522-533, 2019.

[21] Yi N., Xu J., Yan L., and Huang L., "Task Optimization and Scheduling of Distributed Cyber-Physical System Based on Improved Ant Colony Algorithm," *Future Generation Computer Systems*, vol. 109, pp. 134-148, 2020.

[22] Yiqiu F., Xia X., and Junwei G., "Cloud Computing Task Scheduling Algorithm Based on Improved Genetic Algorithm," *in Proceeding of IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference*, Chengdu, pp. 852-856, 2019.

[23] Zhang J., Hu X., Ning Z., Ngai C., Zhou L., Wei J., Cheng J., and Hu B., "Energy-Latency Tradeoff for Energy-Aware Offloading in Mobile Edge Computing Networks," *IEEE Internet of Things Journal*, vol. 5, no. 4, pp. 2633-2645, 2018.

Qingzhu Wang received the B. E., M.E., and Ph.D. degrees from Jilin University, Changchun, China, in 2006, 2008, and 2011, respectively. Shi is currently with the School of Information Engineering, Northeast Electric Power University as an Associate Professor. Her interests are multi-dimensional signal processing.

Xiaoyun Cui was with the School of Computer Science, Northeast Electric Power University, Jilin, China. He is studying for his Master's degree for the field of Computer Science and Technology.