Delay and Energy Consumption Optimization Oriented Multi-service Cloud Edge Collaborative Computing Mechanism in IoT

Sujie Shao[1], Jiajia Tang[1], Shuang Wu[2], Jianong Li[3], Shaoyong Guo[1] and Feng Qi[1,4,*]

[1]*State Key Laboratory of Networking and Switching Technology, Beijing University of Posts and Telecommunications, Beijing 100876, China*
[2]*State Grid Ningxia Electric Power Co. Ltd., Yinchuan, Ningxia 750001, China*
[3]*China Electronics Standardization Institute, Beijing 100007, China*
[4]*Peng Cheng Laboratory, Shenzhen, Guangdong 518000, China*
E-mail: qifeng@bupt.edu.cn
[*]*Corresponding Author*

Received 06 July 2021; Accepted 11 August 2021;
Publication 06 November 2021

Abstract

The rapid development of the Internet of Things has put forward higher requirements for the processing capacity of the network. The adoption of cloud edge collaboration technology can make full use of computing resources and improve the processing capacity of the network. However, in the cloud edge collaboration technology, how to design a collaborative assignment strategy among different devices to minimize the system cost is still a challenging work. In this paper, a task collaborative assignment algorithm based on genetic algorithm and simulated annealing algorithm is proposed. Firstly, the task collaborative assignment framework of cloud edge collaboration is constructed. Secondly, the problem of task assignment strategy was transformed into a function optimization problem with the objective of minimizing the time delay and energy consumption cost. To solve this problem, a task assignment algorithm combining the improved genetic algorithm and simulated annealing algorithm was proposed, and the optimal task assignment strategy was obtained. Finally, the simulation results show that compared with the traditional cloud computing, the proposed method can improve the system efficiency by more than 25%.

Keywords: Cloud-edge collaboration, task allocation, genetic algorithm.

1 Introduction

In recent years, with the rapid development of Internet of Things technology, a large number of terminal devices have emerged, and massive data have flooded into the network. In Cisco's 2018 Annual Internet Report, it is pointed out that by 2023, IoT devices will be the fastest growing category of Internet mobile devices in the world, and its compound growth rate is expected to be 30%. By then, IoT connections will account for half of the global connected devices [1]. The huge increase in the number of IoT terminal devices has resulted in the generation of a large amount of IoT data. The popularity of the intelligent terminal equipment has given rise to a variety of emerging applications, has brought the more complicated business, the application of these emerging in large real-time online games, high-definition video streaming and other computationally intensive and delay sensitive business [2], the business of processing capacity and stability of the network are put forward higher requirements.

In the scenario of Internet of Things, both traditional cloud computing technology and edge computing technology have certain limitations. Cloud computing refers to both the applications delivered as services over the Internet and the hardware and systems software in the data centers that provide those services. The inherent limitation of cloud computing technology is that all data generated by terminal devices must be transmitted to the central cloud for calculation [3]. However, the transmission distance from the intelligent terminal to the central cloud server is relatively long, which is limited by the limited bandwidth, and the long distance transmission often leads to too long response time of applications [4]. Moreover, due to the influx of large amounts of data, network congestion and other problems caused by the increased burden of the core network will also affect the user experience. Therefore, the traditional cloud computing technology cannot meet the business requirements in the Internet of Things scenario. In order to make up for the long transmission delay of cloud computing, edge computing technology is proposed [5]. Edge computing refers to the enabling technologies allowing computation to be performed at the edge of the network, on downstream data on behalf of cloud services and upstream data on behalf of IoT services. Here we define "edge" as any computing and network resources along the path between data sources and cloud data centers [6]. Edge computing technology can obtain idle computing power and storage resources in edge devices close to data sources, which can be used to compute services that require high real-time performance. However, in the business scenario of the Internet of Things, all kinds of terminals of the Internet of Things are sending a large amount of data to the network, and there are great differences in the data types and processing requirements of different businesses. However, due to the limitations of the size and capacity of the edge devices, computing and storage resources are limited. The current requirements for computing power in the Internet of Things scenario have far exceeded the capacity of the edge devices, leading to the increasing load of the network, which is insufficient to handle a large amount of Internet of Things data.

To solve the above problems above, a new computing mode of cloud edge collaborative computing is proposed. On the edge of the cloud side in collaborative technology, computing as cloud between the center and the terminal nodes, the server has been deployed near the Internet of things terminal equipment, thus produced by the terminal equipment do not need to be transfer to center all cloud computing data server, reduced the amount of data in the network, thus effectively alleviate the network congestion problem, In addition, the collaborative computation of tasks by edge devices reduces the computational pressure of the core network [7] and enhances the stability of the network and system.

However, in the scenario of the Internet of Things, the correlation between tasks determines that they cannot be randomly assigned, and different services also have different demands for delay, energy consumption, reliability, etc. And, at the edge of the cloud in collaborative architecture involves many network equipment, the different equipment processing capacity, power consumption each are not identical, therefore different task allocation strategy corresponding to the time delay, power consumption, reliability system cost often is large, so the design task allocation strategy, often need to consider the various factors, on the premise of meet the requirements of different tasks, optimize system performance and reduce system cost.

In order to solve the above problems, a new computing mode of cloud edge collaborative computing is proposed. On the edge of the cloud side in collaborative technology, computing as cloud between the center and the terminal nodes, the server has been deployed near the Internet of things terminal equipment, thus produced by the terminal equipment do not need to be transfer to center all cloud computing data server, reduced the amount of data in the network, thus effectively alleviate the network congestion problem. In this paper, we use the improved genetic algorithm combined with simulated annealing algorithm to solve the task assignment strategy for the target of time delay and energy consumption. Moreover, the collaborative computation of tasks by edge devices reduces the computational pressure of the core network and enhances the stability of the network and system.

2 Related Work

In recent years, with the cloud-edge collaboration technology in the application of Internet of Things and 5G network more in-depth, more and more researchers are also more closely focus on the task collaborative allocation problem in cloud-edge collaboration technology, aims to coordinate the computing resources of edge devices and cloud center, improve the network capacity, and process all kinds of complex tasks in current Internet of Things.

At present, the research goals of task assignment algorithms usually include minimizing system delay and energy consumption, reducing network congestion or ensuring system reliability. In reference [8], considering network congestion and long waiting delay caused by traffic scheduling in the network, as well as different input data generation devices required by different tasks, the author proposed a cooperative assignment strategy of data awareness task of joint task scheduling and network traffic, and adopted the multi-stage greedy adjustment algorithm. By considering the distribution of tasks and the adjustment of network traffic, tasks are allocated to avoid network congestion and improve the efficiency of task execution. Lichao Yang et al. proposed an intelligent optimization algorithm to simulate fish foraging behavior, which minimizes the system energy consumption under the constraints of meeting the computing capability of edge devices in the system and the transmission and computational delay of tasks allocated to different devices [14]. Juan Liu et al. adopted the Markov decision process method and designed an efficient one-dimensional search algorithm by analyzing the average delay and average energy consumption in the task execution process. The algorithm comprehensively considered the queuing situation in the buffer, terminal equipment computing and storage resource load. It provides an effective solution to minimize the delay under the condition of limited power [9]. In reference [10], in order to solve the task load allocation problem in the cloud edge collaborative computing architecture of the Internet of

Things, an efficient algorithm was designed to realize the reasonable task load allocation among local edge devices, adjacent edge devices and central cloud server, and to ensure the minimum energy consumption under a certain delay limitation.

At present, researchers at home and abroad have studied a variety of multi-task coordination allocation strategies in cloud-edge collaboration technology. They consider different dimensions and balance the resources and performance of edge equipment and central cloud server from different perspectives, and strive to reduce the time and energy consumption cost of task computing. However, most of the current researches on task coordination allocation strategy usually only optimize a single system goal, that is, only to reduce the system delay or energy consumption costs as the optimization goal. We can analyze and adjust the weight of the two in the optimization objective function, so that the delay and energy consumption cost can be used as the optimization objective at the same time. Therefore, in view of the above problems, this paper takes time delay and energy consumption as the common optimization goals, and designs a multi-task cooperative allocation algorithm for cloud edge collaboration.

3 System Model

The traditional collaborative task assignment scenario usually adopts a three-tier task assignment architecture composed of the user terminals, the edge devices and the central cloud [11]. The user terminals are usually intelligent terminal devices of the Internet of Things, such as surveillance cameras, thermostat controllers in smart homes and other devices that can generate data but have limited computing power. Edge server is a layer between the user terminal device and the central cloud, which is usually realized by configuring computing and caching servers at wireless access points such as base stations and WiFi access points. The central cloud integrates computing resources through virtualization and distributed technology and has strong computing capacity. Computing tasks that are generated on a user end device can be performed on a user end device, on an edge device, or on a central cloud.

3.1 Task Assignment Scenario and Problem Description

In the scenario proposed in this paper, on the basis of the traditional three-tier task assignment architecture, edge devices are further divided into local edge devices and remote edge devices according to the distance from terminal

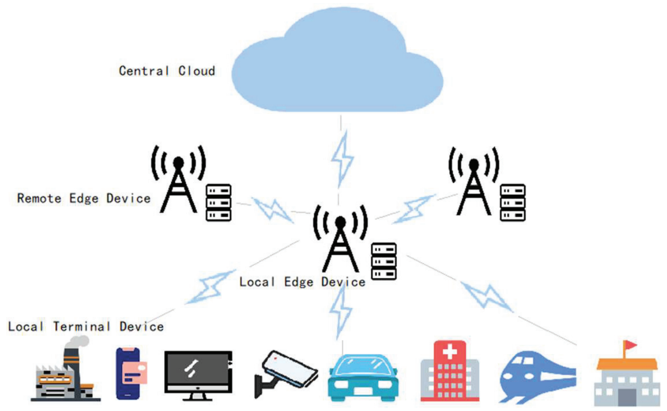

Figure 1 The collaborative task assignment scenario.

devices [12]. The collaborative task assignment scenario of the system is shown in Figure 1. Although the computing capability of the remote edge device is poor compared with that of the central cloud, the transmission delay and energy consumption of the remote edge device are lower than that of the central cloud because the distance between the edge devices is shorter and the energy consumption for data calculation is smaller. On the one hand, this reduces the cost of transferring tasks to the central cloud computing, on the other hand, it also makes full use of adjacent idle devices, reducing the waste of computing resources.

In this scenario, we consider a task assignment scenario consisting of a single local terminal device, a single local edge device, multiple remote edge devices, and a central cloud server. Tasks are generated on a local terminal and can be executed locally or assigned to a local edge device. Tasks on the local edge device may be performed locally or further assigned to the remote edge or central cloud. When tasks are allocated and calculated in this architecture, different task allocation strategies are adopted, so the delay and energy cost generated by the system will be different [13]. Therefore, the problem to be solved by the algorithm in this paper is to find an optimal task allocation strategy in the above scenarios, so as to minimize the overall cost of the system.

3.2 Task Relationship Model

Figure 2 shows the relationship model diagram of the task generated by the user terminal. In this paper, the task relationship model can be described by

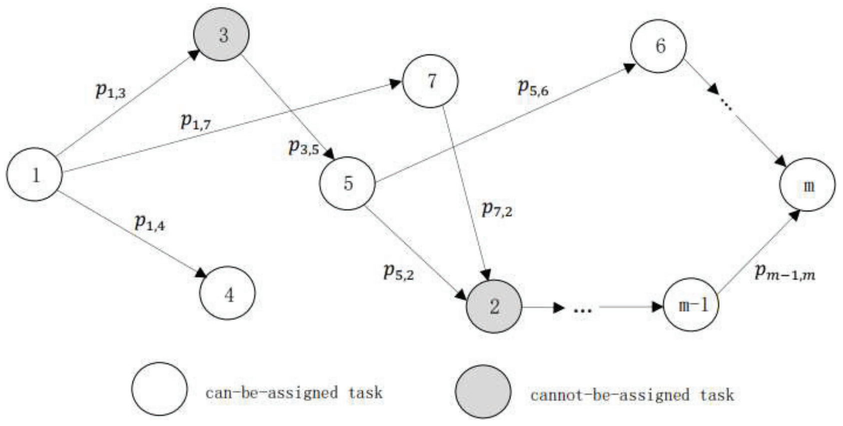

Figure 2 Task relationship model diagram.

a DAG, D = (M, P) [14–16], and each task generated on the user terminal device can be divided into M sub-tasks, that is, $\mathrm{M} = \{1, 2 \ldots, \mathrm{m}\}$, where each subtask corresponds to a vertex in the graph. Each sub-task contains three properties of input data amount (L), calculated data amount (D) and feedback data amount (F). There is interdependences between subtasks. Edge $p_{i,j}$ indicates that the priority of the ith task is higher than that of the jth task. Moreover, part of the tasks can not be assigned to other devices for calculation [17].

3.3 Delay Model

(a) Waiting delay

The waiting time of a task before the computation begins is determined by two factors, the transfer time and the end time of all precursor tasks [18].

It is stipulated that the data transmission rate between edge devices is set as a constant value, R_e. The rates between terminal devices and edge devices, and between edge devices and central cloud are set as variables R_l and R_c respectively. Then the time taken for task m to be transmitted to the local edge is $T^t_{m,le} = L_m/R_l$, and the time taken for transmission to the remote edge and central cloud are $T^t_{m,e} = L_m/R_e + L_m/R_l$ and $T^t_{m,c} = L_m/R_c + L_m/R_l$. Since the task is generated on the terminal device, the transmission delay $T^t_{m,l} = 0$.

On the other hand, due to priority constraints between tasks, the current task must wait for all of its predecessors to complete execution before it can start executing. Therefore, the waiting delay of task m is:

$$T_m^w = max\left(T_m^t, \max_{k\in pre(m)} max\{T_{k,l}, T_{k,le}, T_{k,e}, T_{k,c}\}\right) \tag{1}$$

(b) Computing delay

The data processing capacity of the local terminal device, edge device and central cloud is f_l, f_e, and f_c respectively. Therefore, the calculation delay $T_{m,l}^c = D_m/f_l$ for the task executed on the local terminal device and the calculation delay $T_{m,le}^c = T_{m,e}^c = D_m/f_e$ on the edge device, $T_{m,c}^c = D_m/f_c$ in the central cloud.

(c) Feedback delay

After the task is executed on the assigned device, the result data should be fed back to the user terminal device. Therefore, the feedback delay $T_{m,l}^f = F_m/R_l$ for the calculation result of the local edge equipment, and the feedback delay $T_{m,e}^f = F_m/R_e + F_m/R_l$ for the feedback result of the remote edge equipment, in the same way, $T_{m,c}^f = F_m/R_c$ + F_m/R_l.

(d) Delay optimization objective function

Using matrix $X_{m\times n} = \{x_{ij}|x_{ij} = 0, 1\}$ to represent the task allocation. $x_{ij} = 1$ represents that task i is assigned to the device j, $x_{ij} = 0$ represents task i is not assigned to the device j.

For tasks executed at local terminal, since there is no need to transfer data, the waiting delay before the start of execution is $T_{m,l}^w \geq x_{k1}T_{k,l} + x_{k2}T_{k,le} + x_{k3}T_{k,e} + x_{k4}T_{k,c}$, where $k \in pre(m)$. Also, since the task is created locally and executed locally, the calculated results do not require feedback. Therefore, the total delay of execution of task m on the local terminal device is:

$$T_{m,l} = T_{m,l}^w + T_{m,l}^c \tag{2}$$

For tasks assigned to execute on the local edge device, they take:

$$T_{m,le} = T_{m,le}^w + T_{m,le}^c + T_{m,le}^f \tag{3}$$

For the task assigned to the remote edge device for calculation, They takes:

$$T_{m,e} = T_{m,e}^w + T_{m,e}^c + T_{m,e}^f \tag{4}$$

Similarly, tasks assigned to the central cloud take a similar amount of time to compute on a remote edge device:

$$T_{m,c} = T^{w}_{m,c} + T^{c}_{m,c} + T^{f}_{m,c} \tag{5}$$

Therefore, the total time taken by the system to execute all tasks is the sum of the time taken to execute tasks on the local terminal, local edge, remote edge and central cloud, namely:

$$T_n = \sum_{m=1}^{M} (x_{m1}T_{m,l} + x_{m2}T_{m,le} + x_{m3}T_{m,e} + x_{m4}T_{m,c}) \tag{6}$$

To maximize the task execution efficiency of the system, it is necessary to make the system use the shortest time to complete all tasks. Therefore, the optimization objective function of the system in terms of delay is

$$min \sum_{m=1}^{M} (x_{m1}T_{m,l} + x_{m2}T_{m,le} + x_{m3}T_{m,e} + x_{m4}T_{m,c}) \tag{7}$$

3.4 Energy Consumption Model

(a) Energy consumption of the task

The energy consumption of the equipment in the process of task calculation is usually related to the static rated energy consumption, transmission energy consumption and calculation energy consumption for the execution of the calculation task [19]. Where, The static rated energy consumption of local terminal equipment, edge equipment and central cloud server is static value γ_l, γ_e and γ_c, respectively, and the energy consumed to process the data per unit capacity is k_l, k_e and k_c respectively. In order to simplify the calculation model, in this scenario, the energy consumption generated when the device receives the calculation data is small, which is ignored.

Because the task data is generated on the local terminal device, tasks performed on the local terminal device do not need to take into account the energy consumption generated by data transmission. Therefore, the energy consumption generated in the calculation process of tasks performed on the local terminal equipment is the sum of the static rated energy consumption of the terminal equipment and the calculated energy consumption, namely:

$$E_{m,l} = \gamma_l + D_m * k_l \tag{8}$$

Since the task data executed at the local edge needs to be transmitted from the local terminal, the transmission energy consumption of the task data is $L_m * k_l$. Therefore, for tasks performed on local edge devices, the energy consumption cost generated in the calculation process is the sum of static energy consumption of edge devices, input data transmission energy consumption and calculation energy consumption, namely:

$$E_{m,le} = \gamma_l + L_m * k_l + D_m * k_e \tag{9}$$

The static rated energy consumption and calculation energy consumption of the computing tasks performed on the remote edge devices are the same as those performed on the local edge devices, but the energy consumption of input data transmission also needs to increase the energy consumption of data transmission from the local edge to the remote edge devices. Therefore, the energy consumption of input data transmission for the computing task performed at the remote edge is $L_m * k_l + L_m * k_e$. Therefore, the energy consumption generated by the task performed on the remote edge equipment in the calculation process is:

$$E_{m,e} = \gamma_l + L_m * k_l + L_m * k_e + D_m * k_e \tag{10}$$

Similar to tasks assigned to perform on remote edge devices, the transmission of input data for tasks assigned to perform in the central cloud also goes through two stages, so its data transmission energy consumption is $L_m * k_l + L_m * k_e$. Therefore, the energy cost of tasks assigned to the central cloud is as follows:

$$E_{m,c} = \gamma_c + L_m * k_l + L_m * k_e + D_m * k_c \tag{11}$$

(b) Energy consumption optimization objective function

The energy consumption cost generated by the system's execution of all tasks is the sum of the energy consumption cost generated in the calculation process of all tasks, namely:

$$E_n = \sum_{m=1}^{M} (x_{m1}E_{m,l} + x_{m2}E_{m,le} + x_{m3}E_{m,e} + x_{m4}E_{m,c}) \tag{12}$$

In order to minimize the energy consumption cost of the system during task execution, it is necessary to minimize the energy consumption of the system to complete all tasks, namely:

$$min \sum_{m=1}^{M} (x_{m1}E_{m,l} + x_{m2}E_{m,le} + x_{m3}E_{m,e} + x_{m4}E_{m,c}) \tag{13}$$

3.5 Optimization Problem Model

To maximize the task execution efficiency of the system, the main goal is to minimize the total time spent in the task execution process, so the objective function is

$$min\{\beta Pt + (\beta - 1)Pe\} \tag{14}$$

Wherein, $Pt = T_n/T_{max}$ and $Pe = E_n/E_{max}$.

The system constraint conditions to be satisfied are as follows:

$$x_{i1} + x_{i2} + x_{i3} + x_{i4} = 1 \tag{15}$$

$$x_{ij} \in \{0, 1\} \tag{16}$$

$$\sum_{m=1}^{M} (x_{m,1}T_{m,l} + x_{m,2}T_{m,le} + x_{m,3}T_{m,e} + x_{m,4}T_{m,c}) \leq T_{max} \tag{17}$$

$$\sum_{m=1}^{M} (x_{m,1}E_{m,l} + x_{m,2}E_{m,le} + x_{m,3}E_{m,e} + x_{m,4}E_{m,c}) \leq E_{max} \tag{18}$$

$$T_m^w = max\{x_{k,1}T_{k,l}^c + x_{k,1}T_{k,e1}^c + x_{k,1}T_{k,e2}^c + x_{k,1}T_{k,c}^c, T_m^t\}, k \in pred(m) \tag{19}$$

Among them, constraints (15) and constraints (16) ensure the indivisibility of tasks. Constraint (17) means that the total time taken by the system to complete all computing tasks does not exceed the specified maximum completion time T_{max}; Constraint (18) means that the cost of energy consumption generated shall not exceed the maximum energy consumption E_{max} specified by the system; Constraint (19) ensures that all computation data of the task has been transferred to the assigned execution device before the task begins to execute, and that all precursor tasks of the current task have ended execution.

4 Multi-Task Cooperative Assignment Algorithm Based on Genetic and Simulated Annealing Algorithm

Both genetic algorithm and simulated annealing algorithm are efficient heuristic algorithms. Because the search process of genetic algorithm starts from the whole population, it can compare multiple individuals at the same time, and quickly lock the position interval of the optimal solution, but it is easy to fall into the local optimal solution. Although the simulated annealing algorithm has a slower convergence rate, it can jump out of the local optimum with a certain probability. Therefore, combining the two algorithms and giving full play to their advantages can accelerate the iteration speed and prevent falling into local optimum. The key steps of genetic algorithm and simulated annealing algorithm used in this paper are as follows.

4.1 Genetic Algorithm

(a) Encoding and initialization

To obtain the optimal task co-allocation strategy, this article scenario every possible task allocation strategy as an individual, the genetic algorithm to encode each allocation strategy as a chromosome in genetic algorithm, after each computing tasks in a workflow is assigned the execution of the equipment as a gene in the genetic code.

In the collaborative task assignment scenario proposed in this paper, M sub-tasks of each task division are executed on different devices respectively, so there are M genes on each chromosome. Since each subtask may be assigned to the local terminal, the local edge, the remote edge and the central cloud for computation, each gene may have four values: -2, -1, 0 and 1. To avoid the particularity of individuals in the population, the random method is adopted to generate individuals, and the population size is set as 100.

(b) Fitness function

In this scenario, the smaller the overall cost of the system is, the better the allocation strategy is considered. Therefore, the fitness function takes the reciprocal of the system cost function,

$$f(n) = \frac{1}{\beta Pt + (\beta - 1)Pe} \tag{20}$$

(c) Cross process

Crossover operation is one of the important methods to generate new individuals by genetic algorithm. In this paper, the method of single-point crossover is adopted. According to certain crossover probability q_c, partial sequences of parental chromosomes are exchanged at randomly selected crossover points according to the specified crossover operation rules, so as to obtain two new individuals, which contain some genes from parental chromosomes.

(d) Mutation process

In the collaborative task assignment scenario proposed in this paper, the farther the task execution device is from the terminal device, although the change of delay cost cannot be accurately known, the energy cost must increase due to the increase of static rated energy consumption, transmission energy consumption and calculation energy consumption. The variation process is optimized according to this characteristic. Move the task's computing equipment closer to the terminal. The process is as follows: if the current execution device is a local terminal, it should be executed on the local edge device after mutation; If the current execution device is a local edge, it will be executed at the remote edge after mutation. If the current execution device is at the remote edge, it should be executed in the central cloud after mutation. If the current execution device is the central cloud, it should be executed at the local terminal after mutation.

4.2 Simulated Annealing Algorithm

According to the physical annealing process on the physics and this article task co-allocation scenarios similarity of task assignment problem, in this scenario, the system of the overall cost function was used to simulate the internal energy E, control parameter t to simulate the physical temperature t, gradual attenuation parameter t, and pick up at the termination of the algorithm solution as the optimal solution of the annealing process. The iterative process is as follows: generate a new solution, calculate the difference between the cost function of the current solution and the original solution, judge whether to accept the solution, the value of the attenuation parameter t to judge whether to terminate the algorithm.

The process of generating new solutions is to reverse part of the gene sequence in the original chromosome. Two locations were randomly selected as the starting and ending points of the inverted gene fragments, and the sequence of genes within the starting and ending points was reversed to get a new individual.

4.3 Cloud Edge Collaborative IoT Multi-service Collaborative computing Mechanism Oriented to Time Delay and Energy Consumption Optimization

The algorithm flow chart is shown in the Figure 3.

Step 1: Initialize the population and parameters. Initialize the maximum number of iterations ITER, crossover probability CROSS and so on.

Step 2: Mutate the same individual in the population.

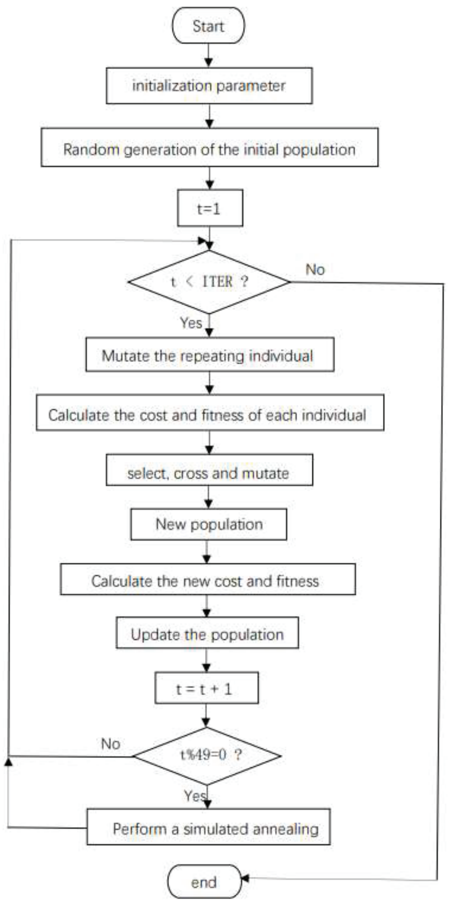
Start
initialization parameter
Random generation of the initial population
t=1
t < ITER ?
No
Yes
Mutate the repeating individual
Calculate the cost and fitness of each individual
select, cross and mutate
New population
Calculate the new cost and fitness
Update the population
t = t + 1
t%49=0 ?
No
Yes
Perform a simulated annealing
end

Figure 3 Algorithm flow chart.

Step 3: Calculated the cost and fitness of each individual, and the roulette wheel method was used to select the individual for crossover or mutation operation.

Step 4: Cross or mutate operations. Cross or mutate the selected individual.

Step 5: Update the population. New individuals are put into the population, and the next generation population is selected from the new population.

Step 6: Simulated annealing. Perform simulated annealing on new population.

Step 7: If the iteration reaches the maximum number of iterations, the solution is considered to be the optimal task cooperative assignment strategy, and the algorithm ends. If the maximum number of iterations is not reached, return to Step 2 and continue.

5 Experimental Simulation and Result Analysis

5.1 Simulation Results

To compare the algorithm performance, four other commonly used algorithms were selected for simulation comparison with the algorithm proposed in this paper (GAMAMACS). These four algorithms are: Traditional cloud-edge collaborative task assignment algorithm, Edge cooperative algorithm [20], Cloud computing algorithm [21], and random assignment algorithm. The comparison results are shown in Figure 4.

As the figure shows, if all tasks are simply transferred to the central cloud, the system cost will be very high. Although the random assignment algorithm can obtain the task assignment strategy with a certain probability to make the system cost less, it cannot guarantee to achieve better results every time due to its large uncertainty. The traditional cloud-edge collaborative task assignment

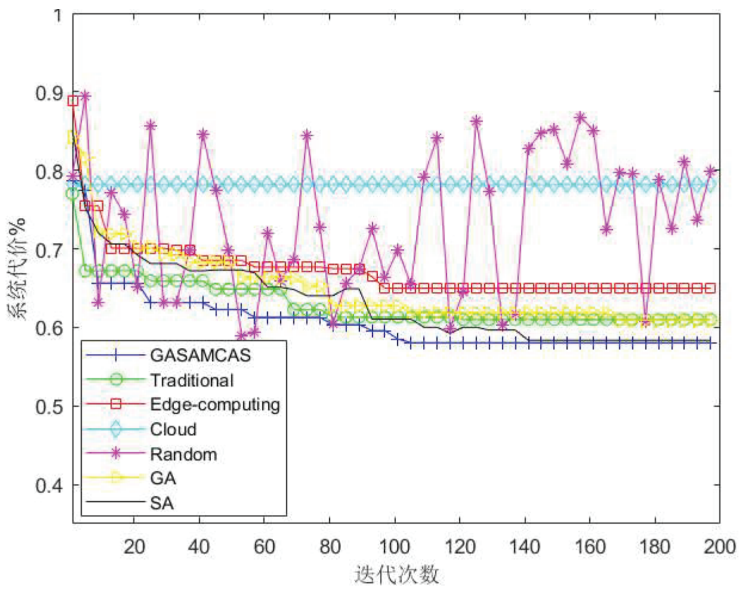

Figure 4 Comparison of simulation results.

algorithm does not consider the remote edge devices, and the multi-edge collaborative algorithm does not consider the edge devices. Therefore, there are fewer computing resources in the network architecture, so the performance is worse than the algorithm used in this paper. The iteration speed of GA is fast, but it may fall into local optimality and the iterative speed of SA is slow. The experimental results show that the combination of the two algorithms can speed up the iteration and avoid falling into local optima. In general, the task collaborative assignment algorithm proposed in this paper can get the minimum overall cost of the system.

5.2 Genetic Algorithm Mutation Probability Value Analysis

The system cost under different values of mutation probability and the optimization of system performance compared with the cloud computing algorithm are shown in Figure 5.

When change of mutation probability values from 0 to 0.02, the system cost is on the decline, the reason is that when mutation probability is low, the probability of introducing new genes in a population is relatively small, its fall into local optimum. The data in the figure shows that too large or too small the mutation probability will lead to the reduction of algorithm performance. Therefore, the mutation probability of genetic algorithm is 0.02 in the process of algorithm simulation.

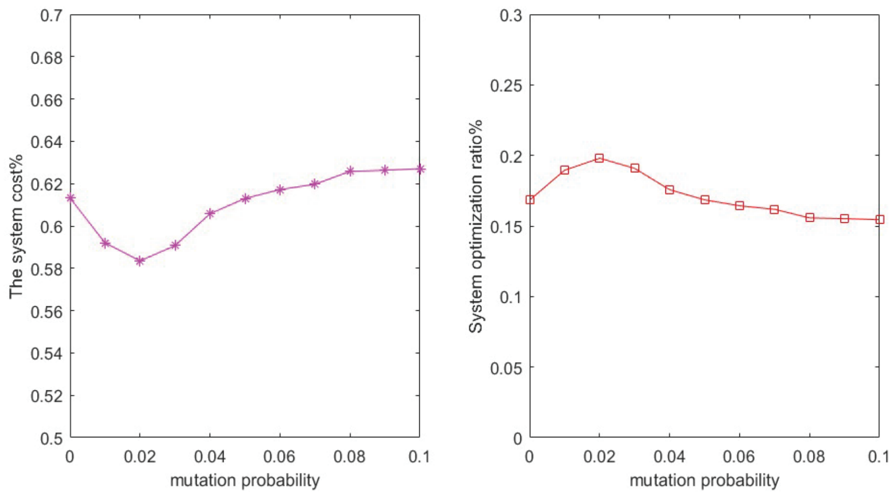

Figure 5 The influence of variation probability value on algorithm performance.

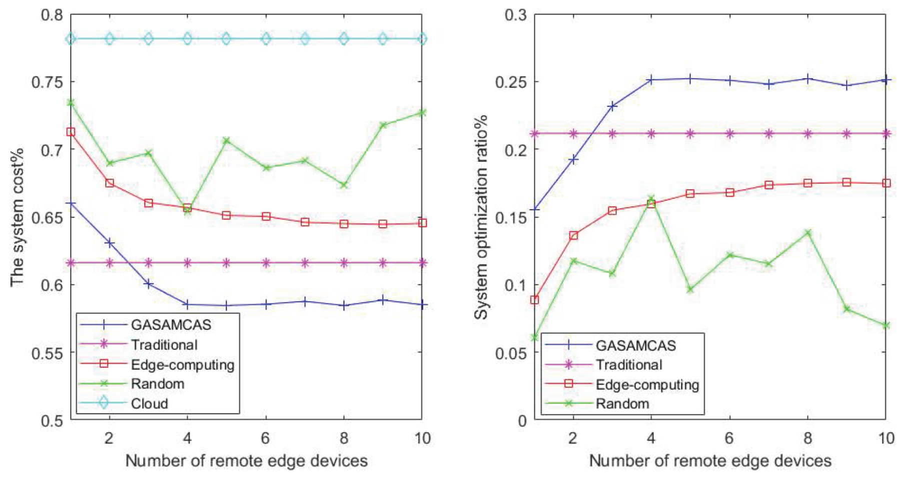

Figure 6 The influence of the number of remote edge devices on algorithm performance.

5.3 Quantitative of Remote Edge Equipment Analysis

Figure 6 compared under the condition of different number of edge equipment, the system cost of different algorithms, and under the value compared to all uploaded to the cloud computing, the optimization of system performance.

In the process of the number of remote edge devices increasing from 1 to 4, the total cost of the system obtained by the algorithm used in this paper also gradually decreases, reaches the minimum value at 4 and gradually becomes stable. Therefore, the number of remote edge devices is set as 4 in the algorithm simulation process.

5.4 The Value of β in System Cost Function Analysis

The influence of the parameter β in the system cost function on the system cost is shown in Figure 7. In this scenario, time delay and energy consumption are two mutually restrictive factors: the allocation scheme with low energy consumption tends to arrange tasks to be executed on the side of terminal equipment, and such allocation strategy usually has a large time delay; However, with a smaller delay, tasks are assigned to different devices, which often means a significant increase in energy consumption. When β is within the range of [0.4, 0.8], although the system cost does not fluctuate much, the delay decreases and the energy consumption increases with the increase of the proportion of delay. In the actual production and life scenarios, in order

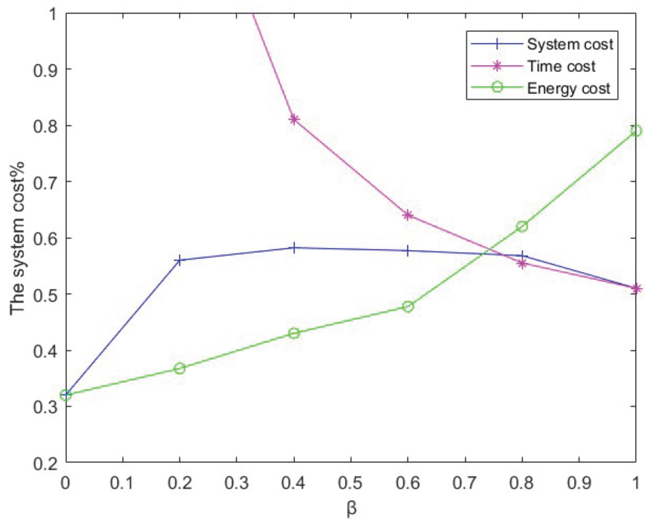

Figure 7 Influence of β on algorithm performance.

to improve the task execution efficiency and shorten the response time, we prefer to exchange relatively greater energy consumption for shorter delay. Therefore, the value of β in the simulation experiment is 0.8.

6 Conclusion

In order to ensure the efficient processing of tasks in cloud edge collaborative architecture, a cloud edge collaborative multi-task collaborative assignment algorithm based on genetic algorithm and simulated annealing algorithm is proposed in this paper. Firstly, the cloud-edge collaboration task assignment scenario was constructed, and the problem was transformed into a functional optimization problem with the objective of minimizing the time delay and energy cost. Secondly, the genetic algorithm is improved and the simulated annealing algorithm is combined to solve the above problems. Finally, simulation results show that compared with the traditional cloud computing, the proposed method can improve the system performance by more than 25%.

Acknowledgement

This work is supported by National Key R&D Program of China (2020AAA 0140102), Key Project Plan of Blockchain in Ministry of Education of the

People's Republic of China (2020KJ010802) and Test bed construction of industrial Internet platform in specific scenes (new mode).

References

[1] CISCO, 'Cisco visual networking index: global mobile data traffic forecast update', 2018–2023 white paper, April, 2018.

[2] Makris N, Passas V, Korakis T, et al. 'Employing MEC in the Cloud-RAN: An Experimental Analysis', EdgeTech@MobiCom, 2018.

[3] Shi W, Dustdar S. 'The Promise of Edge Computing'. Computer, 2016.

[4] Mach P, Becvar Z. 'Mobile Edge Computing: A Survey on Architecture and Computation Offloading'. IEEE Communications Surveys & Tutorials, 2017.

[5] Ren J, He Y, Yu G, et al. 'Joint Communication and Computation Resource Allocation for Cloud-Edge Collaborative System', 2019 IEEE Wireless Communications and Networking Conference (WCNC). IEEE, 2019.

[6] Shi W, Jie C, Quan Z, et al. 'Edge Computing: Vision and Challenges'. Internet of Things Journal, IEEE, 2016.

[7] Carvalho, Glaucio H S, Woungang, et al. 'Analysis of joint parallelism in wireless and cloud domains on mobile edge computing over 5G systems'. Journal of Communications and Networks, 2018.

[8] Sahni Y, Cao J, Yang L. 'Data-Aware Task Allocation for Achieving Low Latency in Collaborative Edge Computing'. IEEE Internet of Things Journal, 2018.

[9] Liu J, Mao Y, Zhang J, et al. 'Delay-Optimal Computation Task Scheduling for Mobile-Edge Computing Systems'. IEEE, 2016.

[10] Guo M, Li L, Guan Q. 'Energy-Efficient and Delay-Guaranteed Workload Allocation in IoT-Edge-Cloud Computing Systems'. IEEE Access, 2019.

[11] Jia M, Cao J, Yang L. 'Heuristic offloading of concurrent tasks for computation-intensive applications in mobile cloud computing'. Proceedings – IEEE INFOCOM, 2014.

[12] Gao J, Wang J. 'Multi-edge Collaborative Computing Unloading Scheme Based on Genetic Algorithm'. Computer Science, 2021.

[13] Wang J, Zhao G, Zhao Z, et al. 'The Optimal Resource Self-configuration Method of Cognitive Network for Survivability Enhancement'. Journal of Web Engineering, 2020.

[14] Sahni Y, Cao J, Zhang S, et al. 'Edge Mesh: A New Paradigm to Enable Distributed Intelligence in Internet of Things'. IEEE Access, 2017.

[15] Yang L, Zhang H, Li M, et al. 'Mobile Edge Computing Empowered Energy Efficient Task Offloading in 5G'. IEEE Transactions on Vehicular Technology, 2018.

[16] Guo H, Liu J. 'Collaborative Computation Offloading for Multiaccess Edge Computing Over Fiber–Wireless Networks'. IEEE Transactions on Vehicular Technology, 2018.

[17] Vu T T, Huynh N V, Hoang D T, et al. 'Offloading Energy Efficiency with Delay Constraint for Cooperative Mobile Edge Computing Networks', GLOBECOM 2018 – 2018 IEEE Global Communications Conference. IEEE, 2018.

[18] Noghani K A, Ghazzai H, Kassler A. 'A Generic Framework for Task Offloading in mmWave MEC Backhaul Networks', GLOBECOM 2018 – 2018 IEEE Global Communications Conference. IEEE, 2018.

[19] Yang L, Zhang H, Xi L, et al. 'A Distributed Computation Offloading Strategy in Small-Cell Networks Integrated With Mobile Edge Computing', IEEE/ACM Transactions on Networking, 2018.

[20] Cerutti G, Prasad R, Brutti A, et al. 'Compact recurrent neural networks for acoustic event detection on low-energy low-complexity platforms'. IEEE Journal of Selected Topics in Signal Processing, 2020.

[21] Badri H, Bahreini T, Grosu D, et al. 'Energy-Aware Application Placement in Mobile Edge Computing: A Stochastic Optimization Approach'. IEEE Transactions on Parallel and Distributed Systems, 2020.

Biographies

Sujie Shao received the Ph.D. degree from the Beijing University of Posts and Telecommunication, Beijing, China, in 2015. He is currently a Lecturer with the State key Laboratory of Switching and Networking Technology of Beijing University of Posts and Telecommunication. His research interests include edge computing, the Internet of Things, smart grids, and communication network management.

Jiajia Tang received the bachelor's degree in Computer Science and Technology from Beijing University of Posts and Telecommunications in 2021. She is currently pursuing the master's degree at the Department of Computing, Beijing University of Posts and Telecommunications. Her main research interests include edge computing and the Internet of Things.

Shuang Wu was born in October 1985. He received the Bachelor's degree in Communication Engineering from Beijing University of Posts and telecommunications, Beijing, China, in 2007. From 2015 to 2020, he was a department head of State Grid Ningxia Information & Telecommunication Company. Since 2020, he has been a vice general manager of State Grid Ningxia Information & Telecommunication Company, China. His current research interests include Information and communication management Internet of Things, Smart Grid and Dispatching management.

Jianong Li, born on February 24, 1990, graduated from University of New SouthWales, master degree of telecommunications. Currently working as standard development engineer in China Electronics Standardization Institute. Committed to the standardization and industry research in the field of blockchain. Main author of White Paper on China Blockchain Technology and Application Development (2016) and Research Report on China Blockchain Technology and Application Development (2018), first author of Blockchain Application Use Cases (2020), lead the development of Information technology-Blockchain and distributed ledger technology-Reference architecture, the first national standard of blockchain in China. Long-term participation in international standardization organizations (ISO/TC307, IEEE, etc).

Shaoyong Guo is with the department of State Key Laboratory of Networking and Switching Technology, and received Ph.D. degree at Beijing

University of Posts and Telecommunication. His research interests include Blockchain Application technology, Distributed Intelligence, Edge Computing, Energy Internet, and so on. His main contribution on Industrial Internet Data Sharing theory and technology, including the Sharing Data Complex Connection Relationship Representation Model, Data Sharing Network Resource Collaborative Optimization Mechanism, Cross-Domain Data Security and Trusted Sharing Service Mechanism. He is undertaking many key research and development projects and fund projects, and contributed to a number of pioneering standards proposals in ITU-T. And the systems and devices developed by him have large-scale application. He was awarded the second prize of science and technology progress in Henan and Jiangsu province respectively, the second prize of Science and Technology Progress Award of China Communication Society, and so on.

Feng Qi received the B.S. and the master' degrees from Northeastern University, Shenyang, Liaoning province, China, in 1993 and 1996, respectively.

He is currently a Professor with the Beijing University of Posts and Telecommunications, Beijing, China, where he is involved in scientific research, teaching, and standardization research in information and communications. He authored more than ten ITU-T international standards and a number of industry standards. He has successively served as the Vice Chair of the ITU-T Study Group 4 and Study Group 12. His main research interests include communications software, network management, and business intelligence. Prof. Qi was the recipient of the National Science and Technology Progress Award twice.