IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received July 30, 2020, accepted August 10, 2020, date of publication August 14, 2020, date of current version August 27, 2020.

*Digital Object Identifier 10.1109/ACCESS.2020.3016762*

# Collaborative Optimization Scheduling of Cloud Service Resources Based on Improved Genetic Algorithm

**SHAOJIE LIU AND NING WANG**
School of Information Engineering, Shandong Management University, Jinan 250357, China
Corresponding author: Ning Wang (wangning200658@126.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 61772068, in part by the Shandong Social Science Planning Research Project under Grant 19BJCJ18, in part by the Key Research and Development Plan of Shandong (Soft Science) under Grant 2019RKB01221, in part by a Project of Shandong Province Higher Educational Science and Technology Program under Grant J18KA386, and in part by a Project of Shandong Management University's Scientific Research Sailing Plan under Grant QH2020Z04.

**ABSTRACT** With the wide application of cloud computing technology, the services provided by cloud systems have become increasingly diverse, thus these systems are required to solve tasks of high variety and complexity, with a tremendously extensive amount of task data involved. That is why reasonable scheduling system resources are particularly important in cloud computing research. In this paper, a cloud computing system needs to take into account a wider range of cloud service resource types and collaborative optimization scheduling issues in order to solve the tasks at hand. Firstly, a new adaptive genetic algorithm (NAGA) was proposed. By improving the crossover mutation genetic operator, the algorithm was able to save excellent individuals as much as possible, enhance the algorithm's optimization ability, and greatly reduce the probability of the algorithm falling into the local optimal solution. Secondly, focusing on the main factors affecting service quality, such as task completion time, system load, and network bandwidth, an upgraded fitness operator method for the cloud resource collaborative optimization scheduling problem is set forth. Finally, an algorithm of cloud service resources based on an improved genetic algorithm (OSIG) is proposed. Experiments on the CloudSim cloud computing simulation platform demonstrate that the OSIG algorithm proposed in this paper can effectively optimize the resource scheduling strategy, shorten the task completion time, facilitate the system load balancing, and boost the system's service quality. The theoretical analysis was consistent with the experimental results.

**INDEX TERMS** Cloud computing, resource scheduling, genetic algorithm, quality of service.

## I. INTRODUCTION

Given the progressively rising amount of digital resources being deployed in cloud systems, the corresponding resource optimization scheduling mechanism has become a key technology that restricts its further promotion. Aiming to fulfill the need for the cloud computing system to take into account a wider range of cloud service resource types and collaborative optimization scheduling, study the security, reliability and scalability of cloud computing architectures, management mechanisms, and cloud service resources that can carry more resource models, the modeling method, collaborative optimization of cloud service resources, and dynamic optimization scheduling algorithm of cloud resources, etc. have become the center of interest in current research projects. Improving the utilization rate of computing resources, enabling tasks to be completed in a shorter time, reducing the response time of cloud computing services, and improving the quality of services are the main goals of cloud computing resource scheduling which are mainly accomplished by reasonably allocating tasks to different computing resources.

Due to the principle of pay-as-you-go for cloud platforms, users hope that their computing tasks can be completed in less time, while enabling their storage data to occupy as little space as possible. Meanwhile cloud computing platform operators hope to use less computing resources to solve more

The associate editor coordinating the review of this manuscript and approving it for publication was Francisco Rafael Marques Lima.

users' computing and storage tasks [1]. By assigning computing tasks of different users to different computing resources in a scientific and reasonable manner, computing resources with a strong computing power are capable of completing more tasks [2]. An efficient resource scheduling strategy can not only reduce the users' task completion time, but also allow for a rational consumption of the operator's resources, thus avoiding wastage of the computing resources [3]. As an NP problem, cloud resource scheduling can be solved by heuristic algorithm search [4].

This paper's main contributions are as follows: (1) To improve the adaptive crossover and mutation operators in genetic algorithms, a new adaptive genetic algorithm (NAGA) is proposed. (2) Regarding the cloud resource cooperative optimization scheduling problem, this paper takes the task execution time, computing resource load and task transmission time as the bases to evaluate the individual population, and puts forward an improved method of fitness operator for the cloud resource cooperative optimization scheduling problem. (3) Considering the quality of service of the system, a cloud service resource cooperative optimal scheduling algorithm (OSIG) based on an enhanced genetic algorithm is proposed.

## II. RELATED WORK

Reasonable allocation of computing resources has become the key to solving resource scheduling problems. Many scholars at home and abroad have applied intelligent optimization algorithms to cloud resource scheduling. Among them, Agarwal and Srivastava [5] implemented the particle swarm optimization (PSO) method to improve cloud computing resource scheduling strategies, reduce task completion time, and ameliorate service quality. The advantage of this method is that it has a strong global search ability and a fast convergence speed. Its principal disadvantage is that the PSO algorithm is easily falls into local optimality, and the model uses a single target as index for scheduling evaluation, and fails to consider the impact of other factors on the cloud resource scheduling results in many ways. Zhou *et al.* [6] comprehensively considered the impact of processor execution time and transmission time on cloud platform performance, notions they integrated in their new cloud computing energy consumption model, and based on this model, proposed a dynamic adjustment of inertia weight based on the number of iterations. The task scheduling optimization algorithm (M-PSO) is used to optimize the PSO's problems of local optimization and slow convergence speed. Similarly, Jacob and Pradeep [7] proposed a particle swarm optimization algorithm (RTPSO) based on the data locality adjustment function to solve the weight distribution problem of the particle swarm optimization algorithm. In order to reduce the RTPSO's tendency of falling into the local optimal situation, the RTPSO was combined with the bat algorithm (BA), and a new optimization algorithm RTPSO-B was generated. Valarmathi *et al.* [8] combined the cuckoo algorithm (CS) with the particle swarm optimization algorithm (PSO), resulting in a new intelligent optimization algorithm (CPSO). The combination of the two algorithms improved the overall population convergence rate and made the algorithm easy to implement. Mansouri *et al.* [9] established a hybrid cloud task scheduling algorithm known as FMPSO, which is based on the combination of fuzzy systems, and improved the particle swarm optimization algorithm using crossover and mutation operators to solve the PSO's local optimization problem. Kunhambu *et al.* [10] first screened the service application hierarchical clustering algorithm, and then used the PSO algorithm for processing, thereby ameliorating the efficiency of the PSO algorithm. Considering the different influences of various factors in the optimization process, Panwar *et al.* [11] built the TOPSIS-PSO, a cloud computing resource scheduling algorithm based on a combination of TOPSIS and PSO. This scheduling method uses the TOPSIS algorithm to comprehensively evaluate the execution time, completion time and transmission time of the scheduling strategy solved by the PSO, so as to solve the multi-objective scheduling problem in the cloud environment.

Genetic algorithm, as a heuristic strategy optimization algorithm, has captivated the attention of research scholars worldwide, and it has also been widely used in the research of cloud computing task scheduling strategies. Vijay *et al.* [12]–[14] put forward a multi-objective cloud resource optimization scheduling algorithm based on genetic algorithm (SGA), which takes task completion time and transmission time as excellent evaluation bases for scheduling strategies to minimize the task completion time and transmission time, define the fitness operator as the optimization purpose, and prove the effectiveness of the algorithm through simulation experiments. Zheng *et al.* [15] proposed a resource management framework (RPMGA-RMF) based on the combination of resource prediction and a multi-objective optimization genetic algorithm, thus introducing a multi-objective genetic algorithm (GA) based on the hybrid group coding algorithm to optimize the task scheduling strategy, So as to achieve the adaptive and optimized configuration management of resources. Yiqiu and Junwei [16] presented an adaptive genetic algorithm based on binary coding chromosomes on the basis of adaptive genetic algorithm, and applied the algorithm to cloud computing task scheduling, aiming to shorten the task execution time and reduce the associated cost, hence proving the effectiveness of the algorithm through experiments. Ye *et al.* [17] came up with a cloud computing scheduling optimization model that comprehensively considered task execution time and transmission time, and proposed a hybrid scheduling method based on the combination of the batch processing strategy and genetic algorithm based on this model, which also considered tasks. Priority assignment rules can effectively solve the instance-intensive workflow scheduling problem in a private cloud environment.

At the same time, a gradually growing number of intelligent optimization algorithms have also been applied to cloud computing resource scheduling. Sha *et al.* [18], [19]

suggested that discrete bacterial foraging algorithms could be applied to cloud computing resource scheduling to improve the resource's performance, consumption and migration count. Kimpan and Kruekaew [20] combined the artificial bee colony intelligent optimization algorithm with the heuristic scheduling algorithm, and proposed an artificial bee colony heuristic scheduling algorithm (HABC), which comprehensively considers the execution time and load balancing factors for cloud platform performance Impact, applicable to cloud computing virtual machine resource scheduling in heterogeneous environments. In order to enhance the efficiency of cloud data center resource utilization, Barlaskar *et al.* [21] proposed an enhanced cuckoo search algorithm (ECS) and verified the effectiveness of the proposed algorithm through the Cloudsim simulation platform. Kumar *et al.* [22] released a new load scheduling technology, namely the hybrid genetic gravity search algorithm (HG-GSA), to reduce the execution and transmission costs. This technology is essentially based on the hybrid crossover technology and gravity search algorithm, used to search for particles in the best location in the search space. Abdel-Basset *et al* [23] applied Lévy to the Whale Optimization Algorithm (WOA), proposed an improved intelligent optimization algorithm ILWOA, and combined the improved whale optimization algorithm with bandwidth allocation based on the best-fit strategy. The combination of strategy (BWAP) was used to solve the allocation problem of cloud system virtual machine bandwidth. AVS *et al.* [24] stipulated that the improved Kmeans clustering algorithm could be applied to cloud computing, based on the task length, task priority, deadline and cost as the influencing factors of task classification, in order to classify tasks and virtual machines, thereby improving the performance and efficiency of cloud computing. Xingjun *et al.* [25] designed a task scheduling algorithm based on Gray Wolf Optimization Algorithm (GWO), and aimed at reducing the response time and load, in order to achieve multi-objective optimization based on fuzzy logic.

The combination of multiple optimization algorithms and the sharing of population information can greatly improve the algorithm's iterative efficiency. Yuan *et al.* [26] combined the probabilistic technique known as simulated annealing with particle swarm optimization to produce a new spatio-temporal task scheduling algorithm (STTS), which minimizes task delay and energy cost as optimization goals, and effectively schedules tasks to achieve Meet delay requirements while minimizing energy costs. Ghahramani *et al.* [27] analyzed the technology related to resource allocation in the cloud environment and divided them according to the type of function in order to better understand the Qos problem. Abdullahi *et al.* [28] proposed a chaos symbiotic biological search (CMSOS) algorithm for solving multi-objective large-scale task scheduling optimization problems in IaaS cloud computing, using the chaos optimization strategy to generate the initial population, and the sequence components are replaced by chaotic sequences to increase the diversity among groups by randomizing the SOS phase.

**TABLE 1. Cloud computing resource scheduling algorithm comparison.**

| References | Methodology | Advantages | Disadvantages |
|---|---|---|---|
| [5] | Cloud resource scheduling algorithm based on particle swarm optimization algorithm | ·Fast convergence<br>·Strong global search ability | ·Single goal optimization |
| [7] | Cloud resource scheduling algorithm based on particle swarm optimization and cuckoo optimization | ·Multi-objective optimization<br>·Strong search ability<br>·Fast convergence | ·High operating cost |
| [8] | Cloud resource scheduling algorithm based on improved particle swarm and bat algorithm | ·Strong search ability<br>·Fast convergence<br>·Multi-objective optimization | ·High operating cost<br>·High time complexity |
| [11] | Multi-objective cloud computing resource scheduling algorithm based on particle swarm approaching ideal solution ranking method | ·Multi-objective optimization<br>·Strong global search ability<br>·Multi-factor comprehensive evaluation | ·Easy to fall into the local optimum |
| [12] | Multi-objective cloud resource scheduling algorithm based on genetic algorithm | ·Multi-objective optimization<br>·Strong search ability<br>·The algorithm is easy to implement | ·Easy to fall into the local optimum<br>·High time complexity |
| [23] | Cloud resource scheduling algorithm based on the combination of Lévy and whale optimization algorithm | ·Strong search ability<br>·Multi-objective optimization | ·Single goal optimization |
| [24] | Cloud resource scheduling algorithm based on improved Kmeans clustering algorithm | ·Considering multiple factors<br>·Operating cost bottom<br>·The algorithm is easy to implement | ·Allocation strategy is not as good as intelligent optimization algorithm |

## III. PROBLEM DESCRIPTION AND THEORETICAL BASIS

### A. PROBLEM DESCRIPTION

The cloud computing system is composed of thousands of computing nodes. The storage capacity, data transmission rate, and computing speed of each cloud node are different. A reasonable scheduling strategy is the key to integrating the overall system resources. Moreover, determining the allocation between user tasks and cloud service resources is the main issue encinctured in resource optimization scheduling within cloud systems. On the one hand, the computing speed

of available computing nodes limits the ability of the nodes to perform tasks. On the other hand, the bandwidth load of the nodes has an important impact on the transmission of computing tasks, and even leads to prolonged response time, reduced service quality of the cloud platform, and also blindly expands the bandwidth, resulting in a phenomenon whereby the service cost of the cloud platform is too high. This article considers coordinating the optimization of system computing resources and the load of system service resources while satisfying user service requests, so that in the case of multiple concurrent tasks and large workloads in the cloud platform, the system can still explore a better resource allocation program to improve the system's service quality, enhance the user's experience, and ensure the availability and reliability of the cloud platform.

Due to the uneven quality and performance of nodes in cloud computing, in order to ensure the quality of service and service cost, the main problems studied in this paper include: (1) How to improve the algorithm's solution performance. (2) How to maintain the load balance of nodes in cloud computing systems and reduce the completion time of tasks. (3) How to integrate algorithm optimization parameters to diminish the service-related costs while improving the service quality and other issues.

### B. CLOUD COMPUTING SYSTEM RESOURCE SCHEDULING MODEL

The cloud computing system resource scheduling process is illustrated in Figure 1.

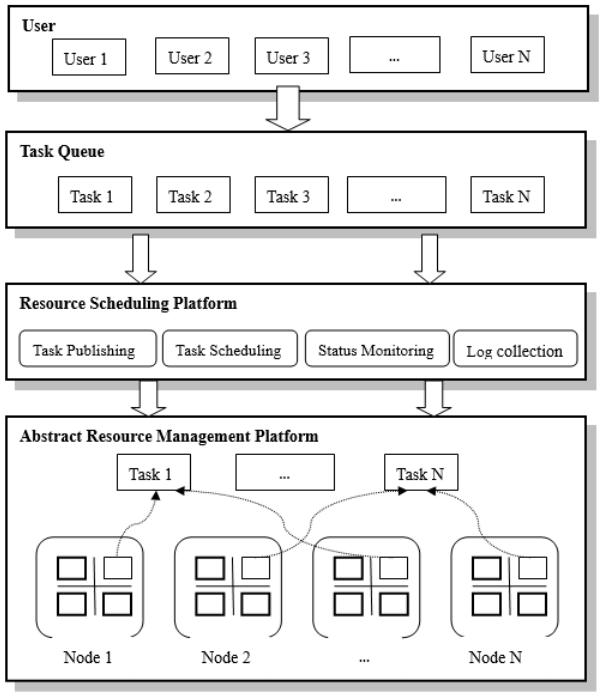



**FIGURE 1.** Cloud computing system resource scheduling process.

The scheduling model of the cloud computing system can be simply described as the first-level mapping relationship $S_1 = \{T, V, f_1\}$, where $T = \{T_1, T_2, \ldots, T_m\}$ represents a set of m tasks. Whereas $V = \{V_1, V_2, \ldots, V_n\}$ represents the set of $n$ virtual machines corresponding to the task list one by one, and $f_1$ stands for the mapping relationship between tasks and virtual machines. The second level mapping relationship can be expressed as $S_2 = \{V, H, f_2\}$, where $H = \{H_1, H_2, \ldots, H_k\}$ symbolizes a set of $k$ physical machines, and $f_2$ is the mapping relationship between virtual machines and physical machines.

Among them, the task execution time ($ET$) is expressed as $ET = max\{VT\}$, where $VT = \{VT_1, VT_2, \ldots, VT_n\}$ indicates the execution time set of n virtual machines; the task transmission time ($TT$) is presented as $TT = max\{VMTT\}$, where $VMTT = \{VMTT_1, VMTT_2, \ldots, VMTT_n\}$ represents the time taken for $n$ virtual tasks to complete the transmission of the assigned tasks. $Load$ signifies the load of the cloud computing system, $Load = sum\{Load_1, Load_2, \ldots, Load_n\}$, where $Load_i$ stands for the load of each virtual machine. Therefore, the resource scheduling model for the cloud computing system can be simplified to equation (1), where $c_i$ is the weight coefficient.

$$\begin{cases} min\, f(x) = c_1 \cdot (ET + TT) + c_2 \cdot Load \\ c_1, c_2 > 0 \\ c_1 + c_2 = 1 \end{cases} \tag{1}$$

### C. THEORETICAL BASIS

Based on the Simple Genetic Algorithm (SGA), Srinvas and Patnaik [29] improved the cross mutation operator and came forth with an Adaptive Genetic Algorithm (AGA) to adaptively adjust the probability of individual crossover and mutation, as well as to optimize the algorithm's solving ability.

#### 1) GENE CODING AND FITNESS FUNCTION

Gene coding basically consists of identifying different solutions in the problem to be optimized, arrange them in a special way, and merge them in the form of character strings. The sequence of character strings formed by each solution is regarded as a chromosome. The main encoding methods in genetic algorithms are binary encoding, floating point encoding, and symbol encoding [30]. Due to the characteristics of the number of tasks and the large number of computing resources in cloud computing resource scheduling, the use of binary encoding would make the encoding too long, and encoding and decoding are more complicated during the evolution process. As a result, this article uses floating point encoding to encode individual populations. The encoding process is as follows:

(1) $Task_i$ randomly assigns a virtual machine computing resource $VM_j$ to each computing task.

(2) The queue of virtual machine computing resource numbers allocated by all tasks is considered as a chromosome, that is, a population individual.

Figure 2 is an example of chromosome coding, the length of the chromosome is the total number of user tasks, whilst the value range of the genes in the chromosome is the number interval of the virtual machine, and the coding of each gene position in the chromosome is the virtual machine number assigned to the task corresponding to the sequence number. The chromosome in Figure. 2 depicts that tasks $T_1$ and $T_4$ are assigned to the virtual machine $VM_1$ for processing, whereas task $T_2$ is processed by the virtual machine $VM_2$, and task $T_3$ is processed by the virtual machine $VM_0$.

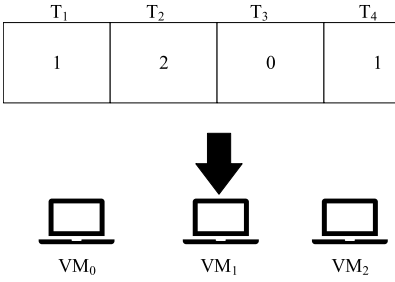



**FIGURE 2.** Examples of chromosome encoding and decoding.

The fitness function is an adequate standard to judge the individual. The fitness function determines the performance of the algorithm to a great extent. It is often designed according to the objective function to solve the problem. The evaluation of individual fitness is generally divided into three processes: decoding the encoded individual to obtain the representation; calculating the individual objective function through the representation; calculating the individual fitness value based on the optimization problem and the objective function.

#### 2) INDIVIDUAL SELECTION

The process of individual selection in genetic algorithm is the simulation of natural selection, promoting the preservation of excellent individuals and the elimination of poor individuals. There are many ways to carry out the process, among which Roulette is the most commonly used method, and the calculation is as follows (2):

$$P_i = \frac{F_i}{\sum_{i=1}^{popSize} F_i} \tag{2}$$

where $P_i$ is the probability that the individual is selected, $popSize$ is the number of individuals in the population, and $F_i$ is the individual fitness value.

#### 3) ADAPTIVE CROSSOVER MUTATION OPERATOR

In genetic algorithm, the crossover process refers to the simulation of the biological sexual reproduction process, and it is the main way of gene recombination [31]. The crossover modes include: single point crossover, double point crossover and multi-point crossover. The single point crossover mode is displayed in Figure 3.

Mutation operation describes the process of gene mutation in the local or individual position of chromosomes during

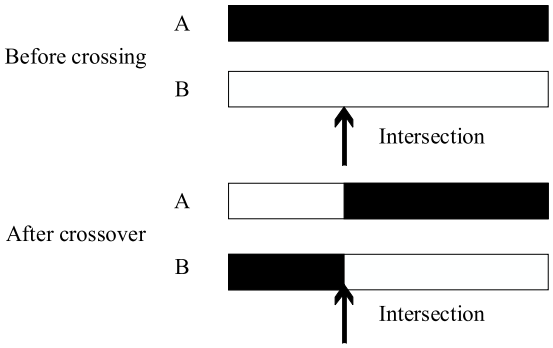



**FIGURE 3.** Single point crossing operation.

biological evolution or reproduction. The AGA improves the original equal probability crossover mutation operator and introduces an adaptive crossover mutation operator, as delineated in formula (3) and (4).

$$P_c = \begin{cases} k_1 \dfrac{(f_{max} - f')}{f_{max} - f_{avg}}, & (f' \geq f_{avg}) \\ k_2, & (f' < f_{avg}) \end{cases} \tag{3}$$

$$P_m = \begin{cases} k_3 \dfrac{(f_{max} - f)}{f_{max} - f_{avg}}, & (f \geq f_{avg}) \\ k_4, & (f < f_{avg}) \end{cases} \tag{4}$$

In the above equations, $P_{\mathbf{c}}$ and $P_{\mathrm{m}}$ are the population crossing rate and variation rate respectively, $k_i$ is a constant between 0 and 1, $f_{max}$ represents the individual with the highest fitness in the current population, $f'$ is the maximum fitness value in the two individuals to be crossed, $f_{avg}$ is the average fitness of the current population, and $f$ represents the fitness value of the individuals to be varied.

## IV. ALGORITHM DESIGN AND OPTIMIZATION

### A. BASIC IDEA BEHIND THE ALGORITHM DESIGN

Improvement of the AGA enables individuals with different fitness values to adaptively adjust the probability of crossover and mutation, which is conducive to improving the fairness of population evolution and improving the algorithm's optimization ability. The variations in the crossover rate and mutation rate are presented in Figure 4 and Figure 5, respectively.

In the process of crossover and mutation, for individuals with high fitness value, the probability of crossover and mutation is reduced, so that the excellent genes are retained. In contrast, for individuals with low fitness value, the probability of crossover and mutation is increased, so that they can explore better gene sequences to a greater extent. Despite the fact that the AGA promotes the evolutionary fairness of the population, in the iterative evolution process of the algorithm, the probability that the population individual is selected is affected by the individual fitness value. The higher the individual fitness value, the greater the probability of being selected. According to the AGA's adaptive cross mutation operator, the probability of cross mutation of the better

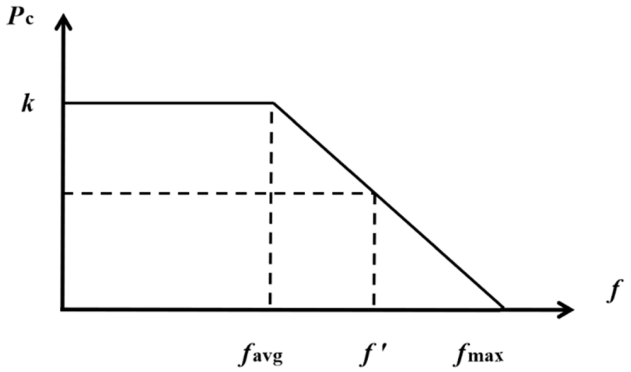



**FIGURE 4.** Trend chart of the crossover rate of AGA algorithm.

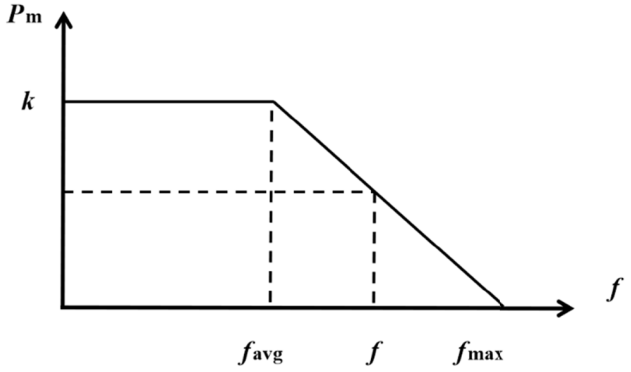



**FIGURE 5.** Trend chart of the mutation rate of AGA algorithm.

individual approaches zero, thus limiting the population of these individuals. The exploration ability of the quality solution can easily be premature, which makes the final solution of the algorithm fall into the local optimum. Based on the above analysis, this paper improves the operator to ensure that individuals get more reasonable probability of crossover and mutation, and enhances the algorithm's overall performance.

### B. IMPROVED ADAPTIVE GENETIC ALGORITHM

#### 1) A NEW ADAPTIVE GENETIC ALGORITHM (NAGA)

On the basis of the AGA, a new adaptive genetic algorithm (NAGA) is proposed in this paper. The cross mutation operator of the new adaptive genetic algorithm is exhibited in formula (5) and (6).

$$P_c = \begin{cases} p_{cmax} - (p_{cmax} - p_{cmin})\dfrac{(f' - f_{avg})}{f_{max} - f_{avg}}, & (f' \geq f_{avg}) \\ p_{cmax}, & (f' < f_{avg}) \end{cases} \tag{5}$$

$$P_m = \begin{cases} p_{mmax} - (p_{mmax} - p_{m\min})\dfrac{(f - f_{avg})}{f_{max} - f_{avg}}, & (f \geq f_{avg}) \\ p_{mmax}, & (f < f_{avg}) \end{cases} \tag{6}$$

In the above equations, $p_{cmax}$, $p_{cmin}$, $p_{mmin}$, $p_{mmax}$ denote the set maximum cross rate, minimum cross rate, maximum variation rate and minimum variation rate, respectively. $f_{max}$ corresponds to the individual with the highest fitness in the current population, $f'$ is the maximum fitness value in the two individuals to be crossed, $f_{\text{avg}}$ is the mean of the current population fitness, and $f$ is the fitness value of the individuals to be varied.

In this paper, the improved crossover mutation operator boosts the rationality of individual crossover mutation rate, which does not only ensure the preservation of excellent individuals, but also enhances the ability of low fitness individuals to explore the optimal solution, and appropriately increases the probability of crossover mutation of the better fitted individuals, so as to prevent the population's evolution from falling into the local optimal solution. The probability changes of crossover and mutation are shown in Figure 6 and Figure 7.

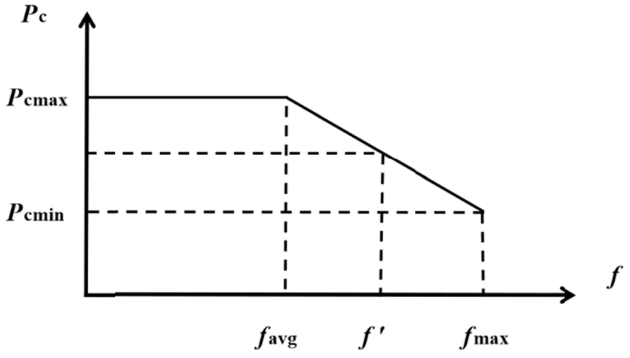



**FIGURE 6.** NAGA algorithm cross rate trend chart.

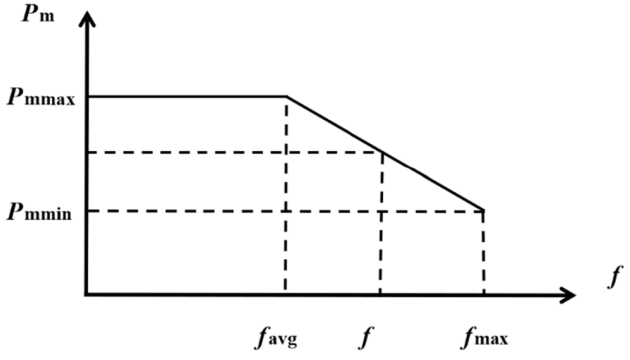



**FIGURE 7.** NAGA algorithm variation rate trend chart.

According to the improved crossover mutation operator, when the individual fitness value increases, the probability of crossover mutation decreases, which enables the algorithm to keep the genes of excellent individuals and make the whole population evolve to a higher fitness level; at the same time, when the individual fitness value gathers near the maximum value, the improved mutation operator can still confer a large mutation rate to the population, breaking the premature state of population evolution to ensure the ability of the population

to explore the optimal solution and optimize the early problem of the AGA algorithm in later evolution stages. A specific description of the NAGA is laid down in Table 2.

**TABLE 2. NAGA algorithm.**

```
Features New Adaptive Genetic Algorithm
Input: popsize, gmax
Output:pop[j]
1.procedure NAGA
2.  pop = Initialize individuals
3.  while i<gmax
4.    for j to popsize
5.      fitness(pop[j])
6.    end for
7.    for j to popsize
8.      Constructing roulette based on fitness values
9.      Selection operation based on improved operator
10.     Cross operation based on improved cross operator
11.     if Offspring> Father
12.       Update schedule
13.     end if
14.     Mutation operation based on improved mutation operator
15.     Update schedule
16.   end for
17.  end while
18.  for j to popsize
19.    if fit(pop[j])>pop[index]
20.      index=j
21.    end if
22.  end for
23.  return pop[index]
24.end procedure
```

#### 2) TIME COMPLEXITY ANALYSIS

The NAGA's complexity analysis is as follows: suppose that the total number of individuals in the population is $n$, and the maximum evolutionary number of the population is $t$, the task size is $l$. From the description of the algorithm in Table 2, it can be seen that the first loop is the iterative process, and the number of executions of the loop is the maximum evolutionary algebra set. In the second loop, individuals of the population perform crossover and mutation operations in each iteration. The number of executions of this loop is the total number of individuals in the population. The third loop is the exchange of individual's genes in the crossover operation, and the number of exchanges is related to the scale of the problem. It can be seen from this that the time complexity of the NAGA algorithm is O($t^*n^*l$).

### C. IMPROVEMENT OF THE FITNESS OPERATOR

This paper provides an explanation of task completion time and system load balancing, and takes them as the main basis of evaluating the resource scheduling effect.

#### 1) TASK COMPLETION TIME

The completion time ($CT$) is an important factor affecting the cloud computing's quality of service. The $CT$ is equal to the sum of the task execution time ($ET$) and the task transfer time ($TT$), as shown in equation (7), while the task execution time is shown in equation (8).

$$CT = ET + TT \tag{7}$$

$$ET = max(VMtime_1, VMtime_2, \cdots, VMtime_m) \tag{8}$$

The $VMtime_i$ is the total time for the ith computing resource to complete the assigned tasks. Since cloud computing task scheduling divides large tasks into numerous small tasks, so as to run in a distributed way, the task execution time is the longest execution time in each computing resource. Calculation of $VMtime_i$ is demonstrated in formula (9):

$$VMtime_i = \frac{instructions_i}{mips} \tag{9}$$

where $instructions_i$ denotes the total number of calculation instructions allocated to the ith calculation resource. $TransitTime$ represents the time spent in task transfer, and the calculation is elicited in equation (10).

$$TT = max(VMTT_1, VMTT_2, \cdots, VMTT_m) \tag{10}$$

In equation (10), $VMTT_i$ is the time consumed by the task transfer of the ith computing resource. The total transfer time is the longest transfer time in each computing resource. Calculation of the transmission time is performed in equation (11).

$$VMTT_i = \frac{2 \cdot Size_i}{DW_i \cdot V} \tag{11}$$

The $Size_i$ is the total amount of task storage allocated to the computing resource, $DW_i$ is the bandwidth of the computing resource, and $V$ is the amount of data that can be transmitted per second under a bandwidth of 1m, expressed in the unit $kb/s$.

#### 2) VIRTUAL MACHINE LOAD

With the dynamic change of task allocation, virtual machine load balancing ($Load$) is the standard to evaluate the balance degree of computing resource allocation, and it is also a crucial part of cloud computing resource scheduling. In this paper, the standard deviation of the completion time of each computing resource task is used as evaluation basis, and the calculation method is expressed in equation (12).

$$Load = \sqrt{\frac{1}{n}\sum_i^n (VMtime_i - \frac{\sum_{j=1}^n VMtime_i}{n})^2} \tag{12}$$

where $VMtime_i$ is the time it takes for each computing resource to complete the tasks it is assigned to.

#### 3) FITNESS OPERATORS FOR CLOUD COMPUTING RESOURCE SCHEDULING

As the decisive factor for population evolution, fitness function directly affects the effectiveness of the algorithm. This paper comprehensively considers the task completion time ($CT$) and the computing resource load balancing ($Load$) in cloud computing as criteria to judge the quality of task

scheduling strategies. The fitness function is displayed in equation (13).

$$F = \frac{1}{c_1 \cdot (ET + TT) + c_2 \cdot Load} \tag{13}$$

In the above eqaution, *ET*, *Load* and *TT* are task execution time, load balance degree and task transfer time of the individual solution, respectively. $c_1$ and $c_2$ are the weights, with the value range being 0-1, and their sum being equal to 1. The setting of their sizes represents the attention to task completion time, load balancing and task transmission time.

In distributed computing, users' tasks are evenly divided into small tasks of the same size, which are allocated to different computing resources. In other words, the execution time of user tasks is the time spent by the longest computing resources to complete the tasks. The load balance (*Load*) is the standard deviation of the completion time of each computing resource task. The smaller the standard deviation, the more balanced the task allocation, thus avoiding the situation where individual computing resources get too busy while other computing resources are idle. User task completion time represents the service quality of the cloud platform, and the time spent in task transmission has a great impact on the overall response time. The fitness operator proposed in this paper considers this factor comprehensively, and takes the transmission time as the principal influencing factor in evaluating the task allocation strategy, which main aim is to reduce the task completion time and improve the strategy's rationality and effectiveness.

### D. CLOUD RESOURCE COLLABORATIVE OPTIMIZATION SCHEDULING ALGORITHM

#### 1) COLLABORATIVE OPTIMIZATION SCHEDULING OF CLOUD SERVICE RESOURCES BASED ON IMPROVED GENETIC ALGORITHM(OSIG)

Based on the NAGA and the improved fitness operator, the OSIG algorithm, a collaborative optimization scheduling algorithm for cloud resources, is proposed. The improved fitness operator with respect to the cloud resource collaborative optimization scheduling problem is applied to the NAGA to increase the iteration efficiency of the solution space optimization capability, while ensuring the superiority of the scheduling strategy in terms of the task completion time and load balancing. The OSIG scheduling algorithm is delineated in Table 3.

**TABLE 3.** OSIG scheduling algorithm.

```
Features Improved cloud computing resource scheduling
Input: CloudLets,Vms
Output: schedule
Parameter: popsize, gmax,c1,c2
1. procedure OSIG
2. schedule = Initialize individuals randomly based on CloudLets and Vms
3. while i<gmax
4.  for j to popsize
5.   Finding fitness values based on fitness functions
6.  end for
7.  for j to popsize
8.   Constructing roulette based on fitness values
9.   Selection operation based on improved operator
10.  Cross operation based on improved cross operator
11.  if Offspring> Father
12.   Update schedule
13.  end if
14.  Mutation operation based on improved mutation operator
15.  Update schedule
16. end for
17. end while
18. for j to popsize
19.  if fit(j)>index
20.   index=j
21.  end if
22. end for
23. return schedule
24. end procedure
```

#### 2) TIME COMPLEXITY ANALYSIS

The OSIG algorithm is a comprehensive strategy to apply NAGA algorithm to cloud computing resource scheduling problem. Suppose the task size is *l*, the maximum evolution algebra of the population is *t*, the number of individuals in the population is *n*, and the number of virtual machines is *m*. In the OSIG algorithm, the first loop is population iteration, the second loop is to operate on individual population, the third loop is to calculate the task assigned to each virtual machine, and the fourth loop is to calculate the execution time, transmission time and load of each virtual machine according to the task allocation, and obtain the individual fitness value. Therefore, the time complexity of OSIG algorithm is O($t^*n^*m^*l$).

## V. DETERMINATION OF ALGORITHM PARAMETERS

### A. ADAPTIVE CROSSOVER MUTATION OPERATOR

In the OSIG algorithm proposed in this paper, the adaptive cross mutation operator is improved through formula (5) and formula (6). The parameters $p_{\mathrm{cmax}}, p_{\mathrm{cmin}}, p_{\mathrm{mmax}}, p_{\mathrm{mmin}}$, in the operator have an important influence on the entire algorithm. The role of $p_{\mathrm{c}}$ and $p_{\mathrm{m}}$ is to ensure the rationality of population crossover and variation, in such a manner that the offsprings exhibit a higher level of adaptability, so as to avoid falling into the local optimal solution. In a word, in the design of crossover mutation operator, not only the cross mutation rate of high fitness individuals, but also the cross mutation rate of low fitness individuals should be considered to break the gene arrangement and enhance the search ability for the optimal solution in space.

### B. FITNESS OPERATOR

In the fitness operator formula (13), $c_1$ ($0 \leq c_1 \leq 1$) denotes the weight of the task execution time and task transmission time, while $c_2$ ($0 \leq c_2 \leq 1$) is the weight of the load balancing, $c_1 + c_2 = 1$.

## VI. EXPERIMENTAL ANALYSIS

In order to analyze the effect of the algorithm in cloud computing resource scheduling, we performed a thorough

comparison of aforementioned OSIG and PSO [5], SGA [12], CPSO [7], and RTPSO-B [8] algorithms.

### A. ENVIRONMENTAL SETTING

The computer used in the experiment is an Intel (R) core (TM) i7-6500U 16G memory windows 10 notebook computer, with a JDK version of jdk1.7.0_25, having a 3.2.5 Maven version, and the CloudSim 4 as cloud computing simulation platform.

### B. SETTING OF EXPERIMENTAL PARAMETERS

Table 4 to table 6 portray the configurations of the experiment's calculation resource and model parameters.

**TABLE 4. Data center parameter setting.**

| Parameter Name | Parameter Type | Set Value |
|---|---|---|
| Data Center Name | String | Datacenter0 |
| Host architecture | String | x86 |
| Operating System | String | Linux |
| Virtual Machine Monitor Type | String | Xen |
| Host List | List | Host |
| Cost Of Processing Resources | Double | 1.5 |
| Memory Usage Fee | Double | 0.01 |
| Use Cost Of External Deposit | Double | 0.0001 |
| Bandwidth Usage Fee | Double | 0.0001 |

**TABLE 5. Host parameter setting.**

| Parameter Name | Parameter Type | Set Value |
|---|---|---|
| Host ID | Int | 0,1,2,3 |
| Computation Speed | Int | 1000,1500,2500,5000 |
| Memory (MB) | Int | 10000 |
| External memory (MB) | Long | 10000 |
| Bandwidth | Int | 1000 |

**TABLE 6. Virtual machine parameter setting.**

| Parameter Name | Parameter Type | Set Value |
|---|---|---|
| Virtual Machine Number | Int | 0-19 |
| CPU Processing Power | Int | 1000,1500,2500,5000 |
| Mirror size (MB) | Int | 1000 |
| Memory size (MB) | Int | 512 |
| bandwidth (MB) | Long | 10，8，6，4，2 |
| CPU Quantity | Int | 1 |
| Virtual Machine Monitor Type | String | *Xen* |

Since the setting of the crossover rate and the mutation rate has a massive impact on the final experimental results, this paper first evaluates the algorithm based on different parameter settings. In order encourage the development of higher exploration abilities in the individuals, $p_{\text{cmax}}$ and $p_{\text{mmax}}$ were set as 1, and the experiments were carried out by adjusting $p_{\text{cmin}}$ and $p_{\text{mmin}}$. The changes in the fitness values under different parameters are illustrated in the Figure 8 shown.

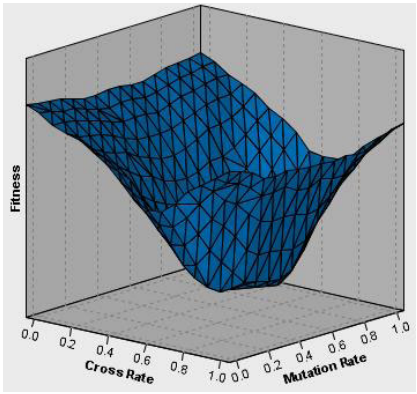



**FIGURE 8. Change trend graph of fitness value.**

According to the experimental results, the algorithm is capable of achieving better results when $p_{\text{cmin}} = 0.6$ and $p_{\text{mmin}} = 0.6$, thus in subsequent experiments, the OSIG parameters were set as demonstrated in Table 7.

**TABLE 7. Parameter setting of OSIG algorithm.**

| Parameter Name | Parameter Type | Set Value |
|---|---|---|
| Population Size | Int | 100 |
| Maximum Evolution Algebra | Int | 250 |
| $p_{cmax}$ | Double | 1 |
| $p_{cmin}$ | Double | 0.6 |
| $p_{mmax}$ | Double | 1 |
| $p_{mmin}$ | Double | 0.6 |
| $c_1$ | Double | 0.5 |
| $c_2$ | Double | 0.5 |

### C. RESULT ANALYSIS

Under the parameter settings of 800 tasks and 10 computing resources, we performed simulation experiments on each algorithm and calculated the execution time, load, and task transmission time of the optimal scheduling strategy obtained by each algorithm in order to analyze the performance of the proposed algorithm.

The comprehensive service quality of the cloud platform is closely related to the task execution time. Reducing the total task execution time can greatly improve the performance of the cloud platform. It can be deduced from Figure 9 that when the number of virtual machines is 10 and the number of tasks is 800, compared with the PSO, CPSO, SGA and RTPSO-B, OSIG algorithms proposed in this paper saves 20.85%, 17.7%, 25% and 12.68% task execution time, respectively. The OSIG algorithm generated better results than other algorithms in the optimization of the task execution time. In the

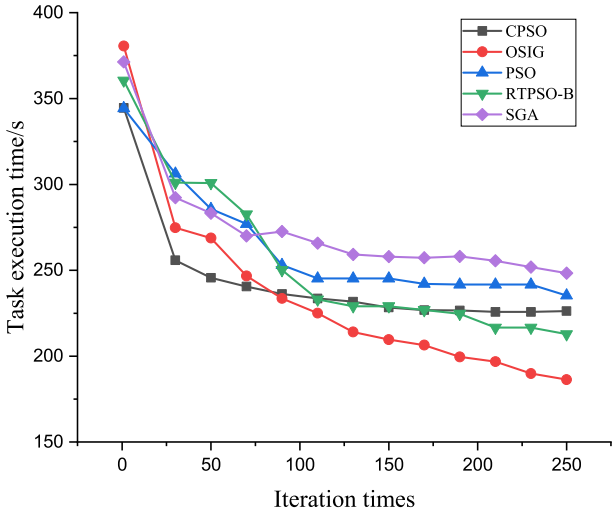



**FIGURE 9.** Task execution time comparison.

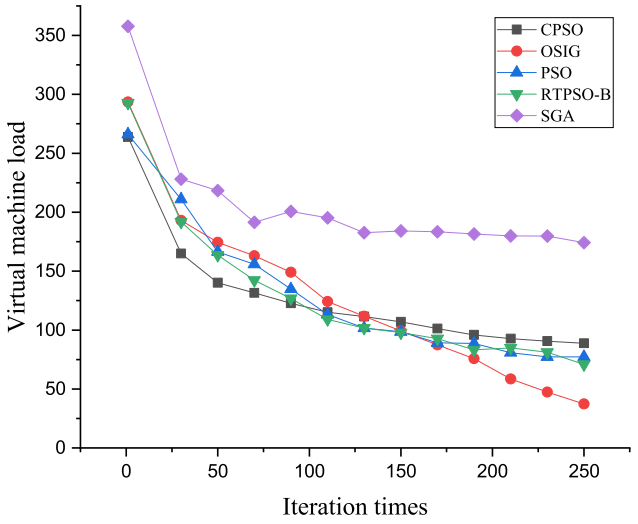



**FIGURE 10.** Comparison of load balancing.

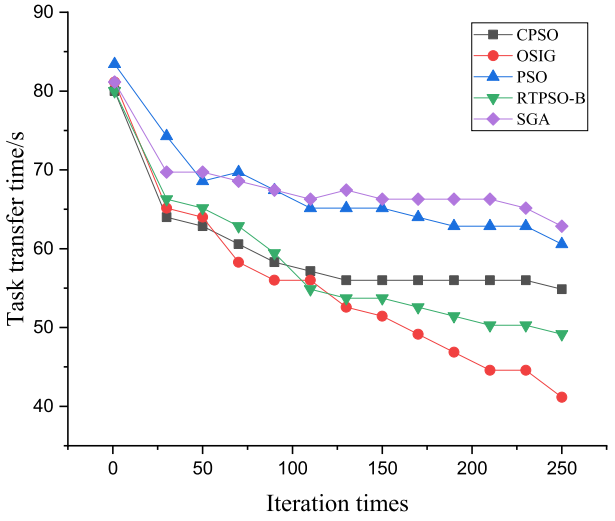



**FIGURE 11.** Task transfer time comparison.

early stage of the iteration, CPSO and RTPSO-B algorithms combining multiple groups can increase the exploration ability of the global solution through the information interaction between the groups, thereby increasing the iteration efficiency. However, in the later stage of the iteration, due to the sharing of group information, the difference between groups becomes smaller, which makes the algorithm easy to fall into local optimum. The cross variation rate in SGA algorithm and the inertia weight in PSO algorithm are fixed values when they are set. Because their values cannot be changed adaptively according to the individual fitness value, the optimization efficiency of the population in the iterative process is limited, so that they fall into local optimization prematurely in the late iteration. During the solution process of the OSIG algorithm, it was able to maintain higher iteration efficiency throughout and break the premature state of population evolution, thereby ensuring the population's ability to explore better solutions. The experimental results were consistent with the theoretical analysis of algorithm design in section 4.2.1, which verifies the improvement rationality of the cross variation operator.

Under the same experimental parameter settings, we calculated the load balance of the scheduling strategy obtained by each algorithm. The results are revealed in Figure 10. Given that the OSIG algorithm proposed in this paper improves the cross-mutation operator, it was equally able to maintain a high exploration ability for more optimal solutions in the later stages of the iteration. Compared with the PSO algorithm, the CPSO, SGA and RTPSO-B algorithms, the OSIG algorithm has significant optimization in load balancing.

Under the setting of the experimental parameters, the task transmission time consumed by the scheduling strategy of each scheduling method is depicted in Figure 11.

The task completion time includes the time during which the user task is sent to the cloud platform. The cloud platform decomposes the large task and transfers it to each computing node and then performs distributed computing. The total time spent during this process is the major factor affecting the service quality, and the task transmission time has a huge impact on this total duration. When the task distribution happens to be uneven, the utilization of node bandwidth is not balanced, which increases the transmission time and affects the overall service. The OSIG algorithm proposed in this paper saves 32.3%, 25.25%, 34.76%, and 16.56% of the task transmission time compared to the PSO, CPSO, SGA, and RTPSO-B algorithms, respectively, hence shortening the task response time, and subsequently improving the service quality.

In the cloud computing environment, the number of tasks and the sizes of computing tasks are huge. By increasing the number of tasks, we can further test the effectiveness of resource scheduling methods. We set the number of tasks to 800 and 1600, respectively. The task execution time consumed by the scheduling method when the number of virtual machines equals 10 is shown in Figure 12.

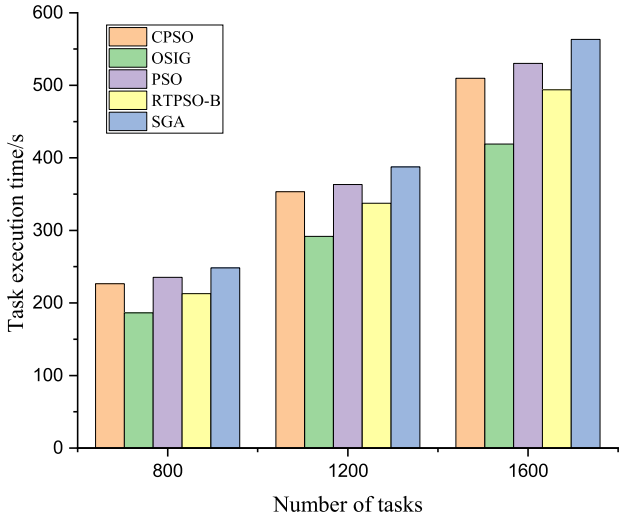



**FIGURE 12.** Change in number of tasks and fitness.

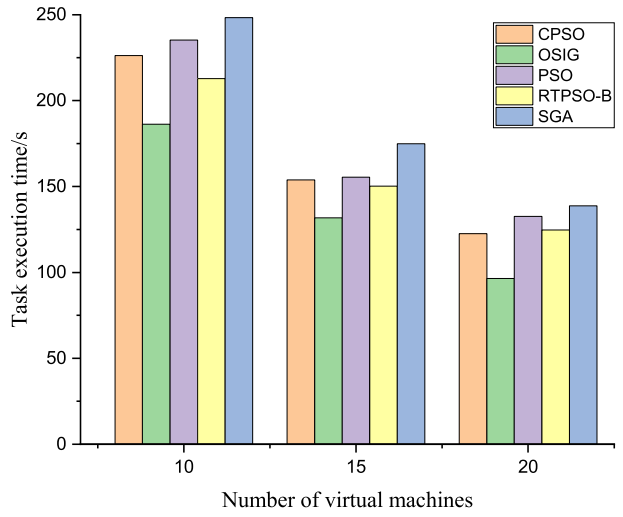



**FIGURE 13.** Number of virtual machines execution time ratio changes.

It can be observed from Figure 12 that the resource scheduling method proposed in this paper becomes more prominent as the number of tasks increases. In a cloud environment with a task volume of 800, the OSIG algorithm proposed in this paper is superior to the improved CPSO and RTPSO-B algorithms comparatively saving up to 17.7% and 12.68% of the task execution time, respectively. Likewise, the OSIG algorithm was able to provide a better scheduling strategy than the traditional SGA. When the number of tasks increased to 1600, the OSIG further reduced the task execution time by 17.87% and 15.21% compared with the CPSO and RTPSO-B algorithms, respectively.

In a cloud computing environment, because the number of tasks and the sizes of the computing tasks are enormous, the number of computing nodes is also large. More computing nodes can effectively reduce the response time of tasks. The number of computing nodes is set to 10, 15 and 20 respectively. The execution time required by the scheduling method is shown in Figure 13.

It can be analyzed from Figure 13 that with the increase in the number of computing resources, the OSIG algorithm is significantly superior to the other resource scheduling algorithms overall, and as the number of computing resources increases, its superiority becomes more prominent. When the number of computing resources is equal to 15 and the amount of tasks is 800, the OSIG algorithm proposed in this paper saves up to 14.37% and 12.66% of task execution time compared to CPSO and RTPSO-B algorithms, respectively. When the computing resources increase to 20, the OSIG algorithm proposed in this paper saves 20.95% and 22.58% of the task execution time compared to the CPSO and RTPSO-B algorithms, respectively.

The experimental results corroborate the effectiveness and reliability of the OSIG algorithm proposed in this paper with respect to cloud computing resource scheduling. Additionally, due to the effective improvement of crossover and mutation operators, the convergence rate of the OSIG algorithm is faster. In addition, with the continuous increase in computing resources, the OSIG algorithm introduced in this paper can more effectively diminish the time spent on task execution, thereby improving the entire system's quality of service and enhancing the user experience. Simultaneously, the optimal allocation strategy of computing resources can be obtained through simulation experiments.

## VII. SUMMARY AND NEXT WORK

First of all, a comprehensive review of the research status of cloud computing resource scheduling models and genetic algorithms was conducted. According to the challenges faced by the existing genetic algorithms, the genetic mutation operator was improved, and a new adaptive genetic algorithm NAGA was proposed, aiming to ensure the convergence speed of genetic algorithms and enhance the exploration ability of better solutions.

Secondly, regarding the problem of cloud resource collaborative optimization scheduling, this paper recognized the task execution time, computing resource load and task transmission time as the bases for assessing the individuals in the population, so that the finally evolved cloud service resource scheduling solution has a shorter task running time and that resources are used more appropriately to avoid the occurrence of overload conditions.

In a nutshell, the cloud resource collaborative optimization scheduling algorithm OSIG based on the NAGA algorithm with an improved fitness operator was proposed. The improved fitness operator based on the cloud resource collaborative optimization scheduling problem was integrated to the proposed NAGA algorithm, which is able to enhance the iteration efficiency of the solution space optimization ability and guarantee the superiority of the scheduling strategy in terms of task completion time and load balancing.

Finally, the algorithm put forward in this paper was implemented in the CloudSim cloud computing simulation platform, and the OSIG algorithm was experimentally compared with the CPSO, PSO, SGA and RTPSO-B algorithms.

Experimental results demonstrated that the OSIG algorithm is significantly superior than other scheduling algorithms.

The NAGA needs to continuously calculate the fitness value of the individual population, which makes the execution process take longer than the SGA. The OSIG algorithm proposed in this paper has a significant effect on cloud resource scheduling compared to other optimization algorithms, but the problem of higher time complexity also exists. We do at the expense of higher iteration time for better cloud resource scheduling strategy. Consequently, we shall explore more efficient and improved methods in future research work, according to the algorithm running time, task size and solution quality, the number of iterations is adjusted adaptively to achieve the balance of the overall time. Moreover, there is a certain difference between the CloudSim simulation environment and reality, and there is also a gap between the amount of data in the simulation and real-time data. Moreover, in the design of the fitness operator, more factors related to cloud resource scheduling should be considered in the future. The dynamic cloud service resource scheduling problem, the priority issue in the process of task calculation and the application of workflow are all our main focus in the next research project.

## ACKNOWLEDGMENT

This research project was conducted by the Key Laboratory of TCM Data Cloud Service of Shandong Management University, the research group on Big-Data-driven abnormal behavior detection technology, and the Ph.D. research fund of Shandong Management University.

## REFERENCES

[1] H. Yuan, J. Bi, and M. Zhou, "Spatial task scheduling for cost minimization in distributed green cloud data centers," *IEEE Trans. Autom. Sci. Eng.*, vol. 16, no. 2, pp. 729–740, Apr. 2019.

[2] A. Wilczyński and J. Kołodziej, "Modelling and simulation of security-aware task scheduling in cloud computing based on blockchain technology," *Simul. Model. Pract. Theory*, vol. 99, Feb. 2020, Art. no. 102038.

[3] G. Zhang and M. N. Ravishankar, "Exploring vendor capabilities in the cloud environment: A case study of alibaba cloud computing," *Inf. Manage.*, vol. 56, no. 3, pp. 343–355, Apr. 2019.

[4] M. Abdullahi and M. A. Ngadi, "Hybrid symbiotic organisms search optimization algorithm for scheduling of tasks on cloud computing environment," *PLoS ONE*, vol. 11, no. 6, Jun. 2016, Art. no. e0158229.

[5] M. Agarwal and G. M. S. Srivastava, "A PSO algorithm based task scheduling in cloud computing," *Int. J. Appl. Metaheur Comput.*, vol. 10, no. 4, pp. 1–17, 2019.

[6] Z. Zhou, J. Chang, Z. Hu, J. Yu, and F. Li, "Amodified PSO algorithm for task scheduling optimization in cloud computing," *Concurrency, Pract. Exper.*, vol. 30, no. 24, pp. e4970.1–e4970.11, 2018.

[7] T. P. Jacob and K. Pradeep, "A multi-objective optimal task scheduling in cloud environment using cuckoo particle swarm optimization," *Wireless Pers. Commun.*, vol. 109, no. 1, pp. 315–331, Nov. 2019.

[8] R. Valarmathi and T. Sheela, "Ranging and tuning based particle swarm optimization with bat algorithm for task scheduling in cloud computing," *Cluster Comput.*, vol. 22, no. S5, pp. 11975–11988, Sep. 2019.

[9] N. Mansouri, B. M. H. Zade, and M. M. Javidi, "Hybrid task scheduling strategy for cloud computing by modified particle swarm optimization and fuzzy theory," *Comput. Ind. Eng.*, vol. 130, pp. 597–633, Apr. 2019.

[10] D. V. Kunhambu and R. V. S. Balan, "Efficient multi-objective particle swarm optimisation based ranking system for cloud service selection," *IET Commun.*, vol. 13, no. 3, pp. 297–304, Feb. 2019.

[11] N. Panwar, S. Negi, M. M. S. Rauthan, and K. S. Vaisla, "TOPSIS–PSO inspired non-preemptive tasks scheduling algorithm in cloud environment," *Cluster Comput.*, vol. 22, no. 4, pp. 1379–1396, Dec. 2019.

[12] J. F. Vijay, "Cloud data analysis using a genetic algorithm-based job scheduling process," *Expert Syst.*, vol. 36, no. 5, p. e12436, Oct. 2019.

[13] P. M. Rekha and M. Dakshayini, "Efficient task allocation approach using genetic algorithm for cloud environment," *Cluster Comput.*, vol. 22, no. 4, pp. 1241–1251, Dec. 2019.

[14] S. Akintoye and A. Bagula, "Improving quality-of-service in cloud/fog computing through efficient resource allocation," *Sensors*, vol. 19, no. 6, p. 1267, Mar. 2019.

[15] S. Zheng, G. Zhu, J. Zhang, and W. Feng, "Towards an adaptive human-centric computing resource management framework based on resource prediction and multi-objective genetic algorithm," *Multimedia Tools Appl.*, vol. 76, no. 17, pp. 17821–17838, Sep. 2017.

[16] F. Yiqiu, X. Xia, and G. Junwei, "Cloud computing task scheduling algorithm based on improved genetic algorithm," in *Proc. IEEE 3rd Inf. Technol., Netw., Electron. Automat. Control Conf. (ITNEC)*, Chengdu, China, Mar. 2019, pp. 852–856.

[17] X. Ye, J. Li, S. Liu, J. Liang, and Y. Jin, "A hybrid instance-intensive workflow scheduling method in private cloud environment," *Natural Comput.*, vol. 18, no. 4, pp. 735–746, Dec. 2019.

[18] J. Sha, A. G. Ebadi, D. Mavaluru, M. Alshehri, O. Alfarraj, and L. Rajabion, "A method for virtual machine migration in cloud computing using a collective behavior-based metaheuristics algorithm," *Concurrency Comput., Pract. Exper.*, vol. 32, no. 2, p. e5441, Jan. 2020.

[19] N. Tao and J. Hua, "A cloud based improved method for multi-objective flexible job-shop scheduling problem," *J. Intell. Fuzzy Syst.*, vol. 35, no. 1, pp. 1–7, 2018.

[20] W. Kimpan and B. Kruekaew, "Heuristic task scheduling with artificial bee colony algorithm for virtual machines," in *Proc. Joint 8th Int. Conf. Soft Comput. Intell. Syst. (SCIS), 17th Int. Symp. Adv. Intell. Syst. (ISIS)*, Aug. 2016, pp. 281–286.

[21] E. Barlaskar, Y. J. Singh, and B. Issac, "Enhanced cuckoo search algorithm for virtual machine placement in cloud data centres," *Int. J. Grid Utility Comput.*, vol. 9, no. 1, pp. 1–17, 2018.

[22] D. Chaudhary and B. Kumar, "Cost optimized hybrid genetic-gravitational search algorithm for load scheduling in cloud computing," *Appl. Soft Comput.*, vol. 83, Oct. 2019, Art. no. 105627.

[23] M. Abdel-Basset, L. Abdle-Fatah, and A. K. Sangaiah, "An improved Lévy based whale optimization algorithm for bandwidth-efficient virtual machine placement in cloud computing environment," *Cluster Comput.*, vol. 22, no. S4, pp. 8319–8334, Jul. 2019.

[24] V. Sharma and M. Bala, "An improved task allocation strategy in cloud using modified K-means clustering technique," *Egyptian Inform. J.*, vol. 2, no. 1, pp. 1–8, Feb. 2020.

[25] L. Xingjun, S. Zhiwei, C. Hongping, and B. O. Mohammed, "A new fuzzy-based method for load balancing in the cloud-based Internet of Things using a grey wolf optimization algorithm," *Int. J. Commun. Syst.*, vol. 33, no. 8, p. e4370, May 2020.

[26] H. Yuan, J. Bi, and M. Zhou, "Spatiotemporal task scheduling for heterogeneous delay-tolerant applications in distributed green data centers," *IEEE Trans. Autom. Sci. Eng.*, vol. 16, no. 4, pp. 1686–1697, Oct. 2019.

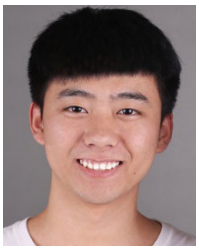
[27] M. H. Ghahramani, M. Zhou, and C. T. Hon, "Toward cloud computing QoS architecture: Analysis of cloud systems and cloud services," *IEEE/CAA J. Autom. Sinica*, vol. 4, no. 1, pp. 6–18, Jan. 2017.

[28] M. Abdullahi, M. A. Ngadi, S. I. Dishing, S. M. Abdulhamid, and B. I. Ahmad, "An efficient symbiotic organisms search algorithm with chaotic optimization strategy for multi-objective task scheduling problems in cloud computing environment," *J. Netw. Comput. Appl.*, vol. 133, pp. 60–74, May 2019.

[29] M. Srinivas and L. M. Patnaik, "Adaptive probabilities of crossover and mutation in genetic algorithms," *IEEE Trans. Syst., Man, Cybern.*, vol. 24, no. 4, pp. 656–667, Apr. 1994.

[30] C. Jatoth, G. R. Gangadharan, and R. Buyya, "Optimal fitness aware cloud service composition using an adaptive genotypes evolution based genetic algorithm," *Future Gener. Comput. Syst.*, vol. 94, pp. 185–198, May 2019.

[31] K. Duan, S. Fong, S. Siu, W. Song, and S. Guan, "Adaptive incremental genetic algorithm for task scheduling in cloud environments," *Symmetry*, vol. 10, no. 5, p. 168, May 2018.



**SHAOJIE LIU** was born in Laiyang, Shandong, in 1997. He is currently pursuing the degree with the School of Information Engineering, Shandong Management University, majoring in information management and information system, with strong hands-on development ability. During the University, he won scholarships for many times. In addition, he actively participated in the research projects of his tutor, published two articles, applied for one invention patent, and won many awards in the national computer related competitions. His main research interests include cloud computing and data mining.

**NING WANG** was born in Yantai, Shandong, in 1978. She received the B.E. degree in computer science and technology from the Shandong University of Technology, Zibo, China, in 2003, the M.Sc. degree in computer software and theory from Yunnan Normal University, Kunming, China, in 2006, and the Ph.D. degree in communication and information system from the University of Science and Technology Beijing, Beijing, China, in 2015. She is currently teaching with the School of Information Engineering, Shandong Management University. Her research interests include cloud computing and big data analysis.